

FSPs der Multi Function Unit (MFU) (ab FW 7.0)

Version vom: Mittwoch, 4. Mai 2022, 12:04:00

Inhaltsverzeichnis

1.	Änderungsliste	1
2.	FSPs MFU	2
	FSP001_ModuleStatus	3
	0x01 _H /1 _D /0x3031 _{ASCII}	
	FSP009_ModuleSerialNumber	5
	0x09 _H /9 _D /0x3039 _{ASCII}	
	FSP010_ModuleCommands	6
	0x0A _H /10 _D /0x3041 _{ASCII}	
	FSP013_PeripheralConfig	7
	0x0D _H /13 _D /0x3044 _{ASCII}	
	FSP014_CurrentScale	8
	0x0E _H /14 _D /0x3045 _{ASCII}	
	FSP015_VoltageScale	9
	0x0F _H /15 _D /0x3046 _{ASCII}	
	FSP016_BFieldScale	10
	0x10 _H /16 _D /0x3130 _{ASCII}	
	FSP020_ActualValue_A	11
	0x14 _H /20 _D /0x3134 _{ASCII}	
	FSP021_ActualValue_B	12
	0x15 _H /21 _D /0x3231 _{ASCII}	
	FSP029_ActualValuePhysicalQuantities	13
	0x1D _H /29 _D /0x3144 _{ASCII}	
	FSP030_SetValue_A	14
	0x1E _H /30 _D /0x3145 _{ASCII}	
	FSP031_SetValue_B	15
	0x1F _H /31 _D /0x3146 _{ASCII}	
	FSP032_SetValue_C	16
	0x20 _H /32 _D /0x3230 _{ASCII}	
	FSP033_SetValue_D	17
	0x21 _H /33 _D /0x3231 _{ASCII}	
	FSP039_SetValuePhysicalQuantities	18
	0x27 _H /39 _D /0x3237 _{ASCII}	
	FSP045_AlteraRemoteUpdateCmd	19
	0x2D _H /45 _D /0x3244 _{ASCII}	
	FSP046_AlteraRemoteUpdateStatus	20
	0x2E _H /46 _D /0x3245 _{ASCII}	
	FSP050_ModuleSupplyValues	21
	0x32 _H /50 _D /0x3332 _{ASCII}	
	FSP053_ModuleTemperatures	22
	0x35 _H /53 _D /0x3335 _{ASCII}	
	FSP054_ModuleTemperaturesComparisonThresholds	23
	0x36 _H /54 _D /0x3336 _{ASCII}	
	FSP058_ParameterChecksumValue	24
	0x3A _H /58 _D /0x3341 _{ASCII}	
	FSP059_ParameterChecksumValueCalculated	25
	0x3B _H /59 _D /0x3342 _{ASCII}	
	FSP060_SlopeLimiter	26
	0x3C _H /60 _D /0x3343 _{ASCII}	
	FSP061_DifferenceCalculatorMultiplier	27
	0x3D _H /61 _D /0x3631 _{ASCII}	
	FSP062_LocalSetValue	28
	0x3E _H /62 _D /0x3632 _{ASCII}	
	FSP063_MPS	29
	0x3F _H /63 _D /0x3346 _{ASCII}	
	FSP064_US1xHS_Multiplexer	30
	0x40 _H /64 _D /0x3634 _{ASCII}	
	FSP065_FrontLemoMultiplexer	38
	0x41 _H /65 _D /0x3431 _{ASCII}	
	FSP066_ModuleInterlockInfosForSCU	40
	0x42 _H /66 _D /0x3432 _{ASCII}	
	FSP067_Defined_US1	41
	0x43 _H /67 _D /0x3433 _{ASCII}	
	FSP068_ButtonAndLEMOInStatus	42
	0x44 _H /68 _D /0x3434 _{ASCII}	
	FSP069_ExternalTriplinesStatus	43

0x45 _H /69 _D /0x3435 _{ASCII}	
FSP070_Controller_1_2_InputSourceSelectionMultiplexer	44
0x46 _H /70 _D /0x3436 _{ASCII}	
FSP071_Controller_1_SetValue	47
0x47 _H /71 _D /0x3731 _{ASCII}	
FSP072_Controller_1_ActualValue	48
0x48 _H /72 _D /0x3438 _{ASCII}	
FSP073_Controller_1_Limits	49
0x49 _H /73 _D /0x3439 _{ASCII}	
FSP074_Controller_1_PI_Settings	50
0x4A _H /60 _D /0x3441 _{ASCII}	
FSP075_Controller_1_I_Part_ComparatorLimits	51
0x4B _H /75 _D /0x3442 _{ASCII}	
FSP076_Controller_1_SetValueDeviation.....	52
0x4C _H /76 _D /0x3443 _{ASCII}	
FSP077_Controller_1_PI_Output.....	53
0x4D _H /77 _D /0x3444 _{ASCII}	
FSP078_Controller_1_P2_Part_ComparatorLimits	54
0x4E _H /78 _D /0x3445 _{ASCII}	
FSP079_Controller_1_SlopeLimiterOutput.....	55
0x4F _H /79 _D /0x3446 _{ASCII}	
FSP081_Controller_2_SetValue	56
0x51 _H /81 _D /0x3531 _{ASCII}	
FSP082_Controller_2_ActualValue	57
0x52 _H /82 _D /0x3532 _{ASCII}	
FSP083_Controller_2_Limits	58
0x53 _H /83 _D /0x3533 _{ASCII}	
FSP084_Controller_2_PI_Settings	59
0x54 _H /84 _D /0x3534 _{ASCII}	
FSP085_Controller2_I_Part_ComparatorLimits	60
0x55 _H /85 _D /0x3535 _{ASCII}	
FSP086_Controller_2_SetValueDeviation.....	61
0x56 _H /86 _D /0x3536 _{ASCII}	
FSP087_Controller_2_PI_Output.....	62
0x57 _H /87 _D /0x3837 _{ASCII}	
FSP088_Controller_2_P2_Part_ComparatorLimits	63
0x58 _H /88 _D /0x3838 _{ASCII}	
FSP089_Controller_2_SlopeLimiterOutput.....	64
0x59 _H /89 _D /0x3839 _{ASCII}	
FSP090_Adder_1_2_SourceSelectionMultiplexer.....	65
0x5A _H /90 _D /0x3541 _{ASCII}	
FSP091_Adder_1_2_Limits.....	67
0x5B _H /91 _D /0x3542 _{ASCII}	
FSP092_Adder_1_SumOut	68
0x5C _H /92 _D /0x3543 _{ASCII}	
FSP093_CorrFactorPI_Limits	69
0x5D _H /93 _D /0x3544 _{ASCII}	
FSP094_CorrFactorPI_kP	70
0x5E _H /94 _D /0x3545 _{ASCII}	
FSP095_ComparatorControl.....	71
0x5F _H /95 _D /0x3546 _{ASCII}	
FSP97_SelVal2CompP2Comp	73
0x61 _H /97 _D /0x3631 _{ASCII}	
FSP098_Selectable_kIkP1.....	75
0x62 _H /98 _D /0x3632 _{ASCII}	
FSP099_Selectable_kIkP1Thresholds.....	76
0x63 _H /99 _D /0x3633 _{ASCII}	
FSP100_V5_ComparatorLimits.....	77
0x64 _H /100 _D /0x3634 _{ASCII}	
FSP101_Degauss_ComparatorLimit.....	78
0x65 _H /101 _D /0x3635 _{ASCII}	
FSP102_PWM_FDrive1_ComparatorLimits.....	79
0x66 _H /102 _D /0x3636 _{ASCII}	
FSP103_PWM_FDrive2_ComparatorLimits.....	80
0x67 _H /103 _D /0x3637 _{ASCII}	
FSP104_CorrFactor_Selector	81
0x68 _H /104 _D /0x3638 _{ASCII}	

FSP105_IGBT_AlternateSetValue	82
0x69 _H /105 _D /0x3639 _{ASCII}	
FSP106_EnergyRecoverLimitation_CurrentDriveValue	83
0x6A _H /106 _D /0x3641 _{ASCII}	
FSP107_DCCT_AdjustmentFactors	84
0x6B _H /107 _D /0x3642 _{ASCII}	
FSP108_CorrFactor_AdderLimits	85
0x6C _H /108 _D /0x3643 _{ASCII}	
FSP109_CorrectionFactorSignals	86
0x6D _H /109 _D /0x3644 _{ASCII}	
FSP110_DACxSourceSelectionMultiplexer FSP110_DACx_and_ScopeChannelx_SourceSelectionMultiplexer	87
0x6E _H /110 _D /0x3645 _{ASCII}	
FSP111_DACGain_Offset	92
0x6F _H /111 _D /0x3646 _{ASCII}	
FSP112_extRAMTriggerStatus	93
0x70 _H /112 _D /0x3730 _{ASCII}	
FSP113_AdderStatus	94
0x71 _H /113 _D /0x3731 _{ASCII}	
FSP114_intScopeTFTSettings	95
0x72 _H /114 _D /0x3732 _{ASCII}	
FSP115_intScopeSourceSelectionMultiplexer	97
0x73 _H /115 _D /0x3733 _{ASCII}	
FSP116_intScopeSettings	100
0x74 _H /116 _D /0x3734 _{ASCII}	
FSP117_intScopeTriggerReadOut	102
0x75 _H /117 _D /0x3735 _{ASCII}	
FSP118_intScopeDataReadOutAddress	103
0x76 _H /118 _D /0x3736 _{ASCII}	
FSP119_intScopeDataReadOut	104
0x77 _H /119 _D /0x3737 _{ASCII}	
FSP120_intFunctionGenerator	105
0x78 _H /120 _D /0x3738 _{ASCII}	
FSP121_ControllerStatusBits LE/SE	107
0x79 _H /121 _D /0x3739 _{ASCII}	
FSP125_LoadSwitchSelection	108
0x82 _H /125 _D /0x3832 _{ASCII}	
FSP229_SW_HighSpeedStream_Synchronized	109
0xE5 _H /229 _D /0x4535 _{ASCII}	
FSP230_SW_intScopeHeaderReadOut	110
0xE6 _H /230 _D /0x4536 _{ASCII}	
FSP231_SW_intScopeDataStreamReadOut	111
0xE7 _H /231 _D /0x4537 _{ASCII}	
FSP232_SW_intSystemParameters	112
0xE8 _H /232 _D /0x4538 _{ASCII}	
FSP233_SW_InterlockTexts	113
0xE9 _H /233 _D /0x4539 _{ASCII}	
FSP234_SW_MDS	114
0xEA _H /234 _D /0x4541 _{ASCII}	
FSP235_SW_Logbook	115
0xEB _H /235 _D /0x4542 _{ASCII}	
FSP236_SW_Delete_Errors	117
0xEC _H /236 _D /0x4543 _{ASCII}	
FSP237_SW_HighSpeedStream	118
0xED _H /237 _D /0x4544 _{ASCII}	
FSP238_SW_SnapshotHighSpeed	119
0xEE _H /238 _D /0x4545 _{ASCII}	
FSP239_SW_Debug	120
0xEF _H /239 _D /0x4546 _{ASCII}	
FSP240_SW_RealTimeClock	121
0xF0 _H /240 _D /0x4630 _{ASCII}	
FSP241_SW_BitManipulation	122
0xF1 _H /241 _D /0x4631 _{ASCII}	
FSP242_SW_CPU_Status	123
0xF2 _H /242 _D /0x4632 _{ASCII}	
FSP243_SW_VerifyHWConfig_ModuleClasses	125
0xF3/243 _D /0x4633 _{ASCII}	
FSP244_SW_ChangeUSIBitrate_ChangeUSIMode	126

0xF4/244 _D /0x4634 _{ASCII}	
FSP245_SW_intScopeDataStream.....	127
0xF5/245 _D /0x4635 _{ASCII}	
FSP246_SW_Recorded_Supplies.....	128
0xF6/246 _D /0x4636 _{ASCII}	
FSP247_SW_Recorded_Temperatures.....	129
0xF7/247 _D /0x4637 _{ASCII}	
FSP248_SW_ReadExtRAMData.....	130
0xF8 _H /248 _D /0x4638 _{ASCII}	
FSP249_Local_Setvalue_Scaling_Factor.....	131
0xF9 _H /249 _D /0x4639 _{ASCII}	
FSP250_NIOS_SW_Version.....	132
0xFA _H /250 _D /0x4641 _{ASCII}	
FSP251_compressed_PCA_configuration_file.....	133
0xFB _H /252 _D /0x4642 _{ASCII}	
FSP252_UpdateMFU_CFI_SoftwareViaRemote.....	134
0xFC _H /252 _D /0x4643 _{ASCII}	
FSP253_UpdateMFU_EPCS_FirmwareViaRemote.....	136
0xFD _H /253 _D /0x4644 _{ASCII}	
FSP254_Parameter_Information_String.....	137
0xFE _H /254 _D /0x4645 _{ASCII}	
FSP255_SW_Flash_VNC2.....	138
0xFF _H /255 _D /0x4646 _{ASCII}	

1. Änderungsliste

Datum	Name	Kommentar
01.07.2021	D. Schupp	Dokument erstellt aus ACU-FSP mUSIc TFT

2. FSPs MFU

Dieses Kapitel behandelt modulspezifische FSPs der MultiFunctionUnit (MFU).

Name	FSP001_ModuleStatus
Adresse	0x01_H/1_D/0x3031_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

[23..22] n.u., immer '0'

[21] Zustand Remote/Local Schalter: '0' ACU System im Local-Betrieb '1', ACU System im Remote-Betrieb

[20] Controller Enabled
wenn '1' ist Regler freigegeben

[19..16] Gerätestatus (4 Bit)

[3..0]	Status
0x0	Kein Status lesbar
0x1	<i>cSTATUSSetDefaults/ cSTATUSWaitForParameters</i> keine definierter Status
0x2	<i>cSTATUSUnitOff</i> Gerät ausgeschaltet
0x3	<i>cSTATUSLoadingBank</i> Bank laden
0x4	<i>cSTATUSSwitchingUnitOn</i> Gerät einschalten
0x5	<i>cSTATUSUnitOn</i> Gerät eingeschaltet
0x6	<i>cSTATUSControllerDisabledByFPGAInternalCause</i> FPGA interne Gründe (des Status erzeugenden Moduls) sperren den Regler
0x7	<i>cSTATUSControllerEnabled</i> Regler freigegeben
0x8	<i>cSTATUSSwitchingUnitOff</i> Gerät ausschalten
0x9	<i>cSTATUSControllerDisabledByCommand</i> Das Kommando <i>cCMDDisableController</i> sperrt den Regler
0xA	<i>cSTATUSControllerDisabledByFPGAExternalCause</i> FPGA externe Gründe (des Status erzeugenden Moduls) sperren den Regler
0xB	<i>cSTATUSResetInterlocks</i>
0xC	<i>cSTATUSMachineProtection</i>
0xD	n.u.
0xE	<i>cSTATUSPowerOnReset</i>
0xF	<i>cSTATUSWhenOthers</i> keine definierter Status

[15..12] Kommando (lokales oder Remotekommand)

[3..0]	Kommando
0x0	<i>cCMDNoAction</i> keine Aktion
0x1	<i>cCMDSwitchUnitOn</i> Gerät einschalten (wenn möglich)
0x2	<i>cCMDSwitchUnitOff</i> Gerät abschalten

0x3	<i>cCMDResetUnit</i> Reset durchführen (z.B. Interlocks)
0x4	<i>cCMDDisableController</i>
0x5	<i>cCMDTriggerSomething</i> hiermit lassen sich Sonderfunktionen in Modulen auslösen

- [11..9] n.u. immer 0
- [8] USIsHighSpeed
wenn ,1' ist mindestens eine der USI im Highspeed Mode
- [7..6] immer 0
- [5] NoInterlocks
wenn ,1' stehen keine Interlocks an
Im Modul sind keine Interlocks gespeichert und es stehen auch keine Interlocks an.
- [4] NoErrors
wenn ,1' ist Modul fehlerfrei
Im Modul sind keine Fehler gespeichert die den Betrieb stören.
- [3] NoWarnings
wenn ,1' ist Modul ohne Warnungen
Im Modul sind keine Warnmeldungen vorhanden die den Betrieb zwar nicht stören aber trotzdem überprüft werden müssten (Details im FSP für die Warnungsbits) z.B. Temperatur zu hoch.
- [2] ModuleReady
wenn ,1' ist Modul betriebsbereit
Das Modul ist voll betriebsbereit, nachdem der Nios2 die Bootsequenz abgearbeitet hat.
- [1] ChecksumOK
wenn ,1' Parameter Checksumme OK
Die Prüfsumme für die Modulparameter ist bestätigt.
- [0] ParametersLoaded
wenn ,1' sind die Parameter geladen
Das Modul hat seine Konfigurationsparameter geladen.
Dies ist dann der Fall, wenn entweder der Nios2 die Parameter aus dem internen Flash geladen hat oder die Parameter über einen PC mittels PCA in die ACu geladen wurden.

Name	FSP009_ModuleSerialNumber
Adresse	0x09_H/9_D/0x3039_{ASCII}
Tiefe	12 Byte / 96 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Der FSP enthält die MFU Serien Nummer

Die Serien Nummer ist über einen One Wire Chip von Dallas/Maxim zu erzeugen, da gewährleistet sein muss das die Serien Nummer weltweit nur einmal vergeben ist.

[95..48] Piggy Seriennummer, sofern vorhanden, andernfalls ,0' (48 Bit)

[47..0] MFU Seriennummer (48 Bit)

Name	FSP010_ModuleCommands
Adresse	0x0A_H/10_D/0x3041_{ASCII}
Tiefe	1 Byte / 8 Bit
I/O	lesen / schreiben
Reset	0x00 _H

Wenn ein Modul Kommandos unterstützt (Einschalten, Ausschalten, Reset usw.) dann werden diese über diesen FSP gesetzt.

Ist die MFU Betriebsart „Remote“ erfolgt die Kommandoübermittlung über das Kontrollsystem-Kommandoregister und die Daten aus diesem FSP werden ignoriert.

Zu beachten ist: die MFU reagiert nur auf Änderungen der Kommandos. Soll also das gleiche Kommando mehrfach hintereinander ausgeführt werden, bedarf es jedes Mal dem Kommando *cCMDNoAction* dazwischen.

[7..4] n.u.

[3..0] Command_USB, diese Kommandos werden z.B. von PowerConfigAdvanced oder ein Terminalprogram über die MFU USB Verbindung gesetzt und steuern die Gerätefunktion, sofern sich die MFU in der Betriebsart „Local“ befindet.

[3..0]	Kommando
0x0	<i>cCMDNoAction</i> keine Aktion
0x1	<i>cCMDSwitchUnitOn</i> Gerät einschalten (wenn möglich)
0x2	<i>cCMDSwitchUnitOff</i> Gerät abschalten
0x3	<i>cCMDResetUnit</i> Reset durchführen (z.B. Interlocks)
0x4	<i>cCMDDisableController</i>
0x5	<i>cCMDTriggerSomething</i> hiermit lassen sich Sonderfunktionen in Modulen auslösen

Name	FSP013_PeripheralConfig
Adresse	0x0D_H/13_D/0x3044_{ASCII}
Tiefe	1 Byte / 8 Bit
I/O	lesen / schreiben
Reset	0x82 _H

Auf dem Modul befindliche Peripherie kann mit diesem FSP konfiguriert werden

- [7] Nach einschalten der Spannungsversorgung wird dieses Bit automatisch auf ,1' gesetzt. Werden Parameter in die MFU geladen (per PC oder durch die MFU selbst) die zur Prüfsummenbildung beitragen sollen, muss dieses Bit gelöscht werden, bevor der erste Parameter übertragen wird. Ist das Laden der Parameter beendet, muss dieses Bit wieder auf ,1' gesetzt werden. Im Anschluss daran wird die Vergleichs-Prüfsumme an „FSP058_ParameterChecksumValue“ gesendet. Die Freigabe des Reglers erfolgt aber nur, wenn die Vergleichs-Prüfsumme auch zu der aus den restlichen Parametern gebildeten Prüfsumme passt.
Das Löschen dieses Bit löst ggf. eine gezielte Reglersperre aus und die „alte“ in der MFU errechnete Prüfsumme wird gelöscht.
Die MFU führt im normalen Betrieb zyklische Zugriffe auf die FSP durch, dies auch während neue Parameter geladen werden. Lesezugriffe auf die FSP sind kein Problem, da die Prüfsumme nur bei Schreibzugriffen aufaddiert wird. Damit diese Zugriffe nicht für eine Verfälschung der Parameter Prüfsumme führen, werden Schreib-Zugriffe seitens der MFU unterbunden solange dieses Bit ,0' ist.
Ggf. ist es sinnvoll, dieses Bit nicht mit einem direkten Zugriff auf FSP013 zu löschen, da hierdurch evtl. auch weitere Bits des FSP013 verändert werden. Soll dieses Bit nur gezielt gelöscht werden, kann auch FSP241 benutzt werden.
- [6..4] n.u.
- [3] Wenn ,0' kann Bit [2] vom Anwender über das Menü NICHT geändert werden, d.h. es ist lokal am Gerät NICHT möglich zwischen Strom- und/oder Feldregelung umzuschalten. Wenn ,1' kann Bit [2] hingegen am Gerät lokal über das Menü geändert werden. Nach dem Reset ist diese Bit immer ,0', d.h. der Nutzer kann lokal den Regelmodus NICHT ändern.
- [2] Wenn ,0' wird Stromgeregelt, wenn ,1' Feldgeregelt. Nach dem ,Reset' ist immer die Stromregelung gewählt.
- [1] Wenn ,0' wird der Regler gesperrt, d.h. für eine Reglerfreigabe muss dieses Bit gesetzt werden. Dieses Bit ist UND-Verknüpft mit Bit[7].
- [0] wenn '1' können über den USB Port (Front MFU) Sollwerte und Kommandos (Ein, Aus, Reset) gesetzt werden, wenn '0' über Backplane-Interface (Interface-Karte oder SCU)

Name	FSP014_CurrentScale
Adresse	0x0E_H/14_D/0x3045_{ASCII}
Tiefe	4 Byte / 32 Bit
I/O	lesen / schreiben
Reset	0x0000_000A _H

Repräsentiert die Stromskalierung des ACU Systems und des darin zur Anwendung kommenden DCCT (10V = CurrentScale Ampere).

[31] wenn ,1' → bipolar, dabei reicht es, wenn FSP14[31], FSP15[31] oder FSP16[31] gesetzt ist, das Gerät wird dann in jedem Fall als bipolar betrachtet

[30..24] n.u.

[23..0] Stromskalierung (24 Bit)

Name	FSP015_VoltageScale
Adresse	0x0F_H/15_D/0x3046_{ASCII}
Tiefe	4 Byte / 32 Bit
I/O	lesen / schreiben
Reset	Reset:0x0000_000A _H

Repräsentiert die Spannungsskalierung des ACU Systems

[31] wenn ,1' → bipolar, dabei reicht es, wenn FSP14[31], FSP15[31] oder FSP16[31] gesetzt ist, das Gerät wird dann in jedem Fall als bipolar betrachtet

[30..24] n.u.

[23..0] Spannungsskalierung (24 Bit)

Name	FSP016_BFieldScale
Adresse	0x10_H/16_D/0x3130_{ASCII}
Tiefe	4 Byte / 32 Bit
I/O	lesen / schreiben
Reset	Reset:0x0000_000A _H

Repräsentiert die Feldskalierung des ACU Systems und der darin zur Anwendung kommenden Feldsonde.

- [31] wenn ,1' → bipolar, dabei reicht es, wenn FSP14[31], FSP15[31] oder FSP16[31] gesetzt ist, das Gerät wird dann in jedem Fall als bipolar betrachtet
- [30..24] n.u.
- [23..0] Feldkalibrierung (24 Bit)

Name	FSP020_ActualValue_A
Adresse	0x14_H/20_D/0x3134_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann der Istwert_A gelesen werden. I.d.R. ist dies der Stromistwert des DCCT.

[23..0] Vorzeichenbehafteter 20 Bit Wert

Name	FSP021_ActualValue_B
Adresse	0x15_H/21_D/0x3231_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann der Istwert_B gelesen werden. I.d.R. ist dies der Feldistwert der Feldsonde.

[23..0] Vorzeichenbehafteter 20 Bit Wert

Name	FSP029_ActualValuePhysicalQuantities
Adresse	0x1D_H/29_D/0x3144_{ASCII}
Tiefe	2 Byte / 16 Bit
I/O	lesen / schreiben
Reset	0x0030 _H

Gibt die physikalische Größe der Istwerte an.

[15..12] physikalische Einheit des Istwertes D

[11..8] physikalische Einheit des Istwertes C

[7..4] physikalische Einheit des Istwertes B

[3..0] physikalische Einheit des Istwertes A

[3..0]	Einheit
0x0	Ampere (A)
0x1	Volt (V)
0x2	Temperatur (C)
0x3	Magnetische Flussdichte (T)
0x4	Magnetische Flussdichte (G)
0x5	n.u.
0x6	n.u.
0x7	n.u.
0x8	n.u.
0x9	n.u.
0xA	n.u.
0xB	n.u.
0xC	n.u.
0xD	n.u.
0xE	n.u.
0xF	n.u.

Name	FSP030_SetValue_A
Adresse	0x1E_H/30_D/0x3145_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x000000 _H

Über diesen FSP wird der Sollwert_A im Remotebetrieb an die MFU übermittelt.

[23..0] Vorzeichenbehafteter 20 Bit Wert (0xFFFF bis 0x7FFF)

Name	FSP031_SetValue_B
Adresse	0x1F_H/31_D/0x3146_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x000000 _H

Über diesen FSP wird der Sollwert_B im Remotebetrieb an die MFU übermittelt.

[23..0] Vorzeichenbehafteter 20 Bit Wert (0xFFFF bis 0x7FFFF)

Name	FSP032_SetValue_C
Adresse	0x20_H/32_D/0x3230_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x000000 _H

Über diesen FSP wird der Sollwert_C im Remotebetrieb an die MFU übermittelt.

[23..0] Vorzeichenbehafteter 20 Bit Wert (0xFFFF bis 0x7FFFF)

Name	FSP033_SetValue_D
Adresse	0x21_H/33_D/0x3231_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x000000 _H

Über diesen FSP wird der Sollwert_D im Remotebetrieb an die MFU übermittelt.

[23..0] Vorzeichenbehafteter 20 Bit Wert (0xFFFF bis 0x7FFFF)

Name	FSP039_SetValuePhysicalQuantities
Adresse	0x27_H/39_D/0x3237_{ASCII}
Tiefe	2 Byte / 16 Bit
I/O	lesen / schreiben
Reset	0x0030 _H

Gibt die physikalische Größe der Sollwerte an.

[15..12] physikalische Einheit des Sollwertes D

[11..8] physikalische Einheit des Sollwertes C

[7..4] physikalische Einheit des Sollwertes B

[3..0] physikalische Einheit des Sollwertes A

[3..0]	Einheit
0x0	Ampere (A)
0x1	Volt (V)
0x2	Temperatur (C)
0x3	Magnetische Flussdichte (T)
0x4	Magnetische Flussdichte (G)
0x5	n.u.
0x6	n.u.
0x7	n.u.
0x8	n.u.
0x9	n.u.
0xA	n.u.
0xB	n.u.
0xC	n.u.
0xD	n.u.
0xE	n.u.
0xF	n.u.

Name	FSP045_AlteraRemoteUpdateCmd
Adresse	0x2D_H/45_D/0x3244_{ASCII}
Tiefe	6/7 Byte / 48/56 Bit
I/O	lesen / schreiben
Reset	Reset:0x00100000_00_0_0_0_0 _H (SE)

Dieser FSP dient als Kommando FSP für die Altera Remote Update Funktion

Imagetyp lesen

Bit[4] = ,0' (Read)

Bit[8] = ,1' (steigende Flanke startet lesen des Imagetyps)

FSP046[1..0] enthält nun den aktuellen Imagetyp.

Imagetyp wechseln

Bit[4] = ,1' (Write)

Bit[12] = ,1' (steigende Flanke wechselt das Image)

Mit Einführung des CycloneV änderte sich dieses FSP inhaltlich leicht.

Die Startadresse wanderte um 4 Bits nach links (beginnt nicht mehr bei Bit 20, sondern erst bei Bit 24) und wird um 4 weitere Bits ergänzt (hat also nun die Breite 32 Bits). (18.12.19 – DS)

Altes Format

[47..44] n.u.

[43..20] Flash Start Address (ab dieser Adresse wird das Image geschrieben)

[19..17] n.u.

Neues Format

[55..24] Flash Start Address (ab dieser Adresse wird das Image geschrieben)

[23..17] n.u.

Gemeinsam unverändert

[16] Reset WD Disable (only for debug)

[15..13] n.u.

[12] Start Write (steigende Flanke an diesem Bit startet die FSM zum Imagetyp-Wechsel)

[11..9] n.u.

[8] Start Read (steigende Flanke an diesem Bit startet die FSM zum lesen des Image-Type)

[7..5] n.u.

[4] Read_n_Write_Enable (muss ,0' sein damit ,Start Read' überhaupt ausgeführt wird, muss ,1' sein damit ,Start Write' überhaupt ausgeführt wird)

[3..2] n.u.

[1..0] Read Source

Name	FSP046_AlteraRemoteUpdateStatus
Adresse	0x2E_H/46_D/0x3245_{ASCII}
Tiefe	10 Byte / 80 Bit
I/O	lesen
Reset	Reset:0x(siehe Beschreibung) _H

Dieser FSP dient als Status FSP für die Altera Remote Update Funktion

[79..72]	ReconfTriggerCondition
[71..69]	Force Osc_int n.u.
[68]	Force Osc_int
[67..44]	Boot Address
[43..41]	Wachdog Enable n.u.
[40]	Wachdog Enable
[39..8]	Wachdog timeout
[7..5]	Cd_early n.u.
[4]	Cd_early, wenn ,1' ist ein gültiges Application-Image an der Bootadresse zu finden
[3..2]	MSM State n.u.
[1..0]	MSM State ('00' = Factory Image, '11' = Application Image)

Name	FSP050_ModuleSupplyValues
Adresse	0x32_H/50_D/0x3332_{ASCII}
Tiefe	16 Byte / 128 Bit
I/O	lesen
Reset	0x(siehe Beschreibung)

Liefert die vorzeichenbehafteten Betriebsspannungen des Moduls. Immer 2 Byte stehen für eine Spannung, d.h. bei 8 Spannungen ist dieses FSP 16 Byte tief. Die Spannungen sind vorzeichenbehaftet und wie folgt sortiert.

- [127..112] vorzeichenbehaftete -12Volt (13 Bit)
- [111..96] vorzeichenbehaftete Pad1 (13 Bit) (floatet, da Pad offen)
- [95..80] vorzeichenbehaftete Volt (13 Bit)
- [79..64] vorzeichenbehaftete Pad2 (13 Bit) (floatet, da Pad offen)
- [63..48] vorzeichenbehaftete 5 Volt (13 Bit)
- [47..32] vorzeichenbehaftete 3,3 Volt (13 Bit)
- [31..16] vorzeichenbehaftete 2,5 Volt (13 Bit)
- [15..0] vorzeichenbehaftete 1,2 Volt (13 Bit)

Die Bitwertigkeiten sind wie folgt:

bei 1V2, 2V5, 3V3 und 5V0 werden die gemessenen Werte direkt mit dem LSB Wert (SUP_LSB_Size = 0.002441) multipliziert und ergeben so die Spannung. Bei -12V0 und 12V0 erfolgt ebenfalls zuerst die LSB Wert Multiplikation und anschließend die Multiplikation mit dem Faktor 11, der dem vorgeschalteten Spannungsteiler entspricht. Die tatsächlich gemessenen Spannungen liegen nämlich bei 1/11 der Originalspannung.

Name	FSP053_ModuleTemperatures
Adresse	0x35_H/53_D/0x3335_{ASCII}
Tiefe	4 Byte / 32 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Je Temperatur 2 Byte ASCII, also je 1 Byte Vorzeichen behaftetes HEX Zeichen.

[31..24] Alarm_Info

[7..0]	Ausgang des Multiplexers
0x00	n.u., Resetzustand
0x01	Device 1 nicht bereit
0x02	Device 2 nicht bereit
0x04	Device 3 nicht bereit
0x08	DeviceSearchRunDone
0x10	Device 1 Grenze überschritten
0x20	Device 2 Grenze überschritten
0x40	Device 3 Grenze überschritten
0x80	Alarm Interrupt wenn Grenze bei einem Device überschritten

[23..16] Sensor 3: Temperatur Modul Mitte (8 Bit)

[15..8] Sensor 2: Temperatur Längsregler (8 Bit)

[7..0] Sensor 1: Temperatur FPGA (8 Bit)

Die Schwellen der Temperaturgrenzen werden im „FSP054_ModuleTemperaturesComparationThresholds“ festgelegt.

Name	FSP054_ModuleTemperaturesComparisonThresholds
Adresse	0x36_H/54_D/0x3336_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x46_4646 _H

Stellt die Vergleichswerte zur Verfügung bei denen die Temperatursensoren Alarm auslösen sollen, sofern die Temperatur überschritten wurde.

Je Temperatur 2 Byte ASCII, also je 1 Byte Vorzeichen behaftetes HEX Zeichen.

Als Standardwert ist 70° Celsius (70_D = 46_H) gewählt.

[23..16] Vergleichswert Sensor 3: Temperatur Modul Mitte (8 Bit)

[15..8] Vergleichswert Sensor 2: Temperatur Längsregler (8 Bit)

[7..0] Vergleichswert Sensor 1: Temperatur FPGA (8 Bit)

Name	FSP058_ParameterChecksumValue
Adresse	0x3A_H/58_D/0x3341_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x00_0000 _H

Repräsentiert die Vergleichs-Prüfsumme der vom Module empfangenen Parameter. Dieser Wert dient zum Vergleich der im Modul errechneten Prüfsumme.

Die Modul-Prüfsumme wird dabei aus den empfangenen Datenbytes durch aufaddieren gebildet und abschließend mit dem Eintrag von „FSP058_ParameterChecksumValue“ verglichen.

[23..0] Prüfsumme der Datenübertragung vom PC zur MFU.

Die Prüfsumme wird im Modul ChecksumBuilder der Teil von mUISC (modular-USI-control) ist aus den Daten der beschriebenen FSP gebildet und abschließend mit dem Wert dieses FSP verglichen.

Name	FSP059_ParameterChecksumValueCalculated
Adresse	0x3B_H/59_D/0x3342_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Repräsentiert die errechnete Prüfsumme innerhalb des Moduls. Gibt die aktuell im Modul errechnete Prüfsumme zurück. Dadurch kann der Fortschritt der Prüfsummenbildung jederzeit verifiziert werden.

[23..0] errechnete Checksumme der Datenübertragung vom PC zur MFU.

Name	FSP060_SlopeLimiter
Adresse	0x3C_H/60_D/0x3343_{ASCII}
Tiefe	30 Byte / 240 Bit
I/O	lesen / schreiben
Reset	0x00_000000_000000_000000_000000_0000_0000_0000_0000_ 745D17_8BA2E8 _H

Repräsentiert Parameter des Steilheitsbegrenzers.

[239] Wenn ,1' → RampRiseTime_E in mV/s

[238] Wenn ,1' → RampRiseTime_D in mV/s

[237] Wenn ,1' → RampRiseTime_D in mV/s

[236] Wenn ,1' → RampRiseTime_B in mV/s

[235] Wenn ,1' → RampRiseTime_A in mV/s

[234] Wenn ,1' → ExpertMode aktive

[233] Aktiviert den Steilheitbegrenzer,
wenn (,1' ODER ,CurrentEqual') UND ,ACU_ControllerEnable'

[232] Wenn ,0' wird der Startwert des Steilheitbegrenzers mit 0 geladen, andernfalls der Wert
des Eingangs ,Dynamic_Threshold_RiseTime_Selection'.
Dies ist i.d.R. ,Controller1_ActValueMuxOut'

[231..216] RampThresholdValue_D (16 Bit)

[215..200] RampThresholdValue_C (16 Bit)

[199..184] RampThresholdValue_B (16 Bit)

[183..168] RampThresholdValue_A (16 Bit)

[167..144] RampRiseTime_E (24 Bit)

[143..120] RampRiseTime_D (24 Bit)

[119..96] RampRiseTime_C (24 Bit)

[95..72] RampRiseTime_B (24 Bit)

[71..48] RampRiseTime_A (24 Bit)

[47..24] Obere Grenze (24 Bit)

[23..0] Untere Grenze (24 Bit)

Name	FSP061_DifferenceCalculatorMultiplier
Adresse	0x3D_H/61_D/0x3631_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x03E8_03E8_03E8 _H

Repräsentiert den Multiplikator für den Multiplikant Delta I der Regelabweichung der Regler 1 und 2, bzw. den Multiplikant Delta I der Differenzen zwischen Istwert Regler 1 und Istwert Regler 2.

[47..43] n.u.

[42..32] (bis MFU_SE_7_2) Multiplikator (11 Bit) für den Differenzbilder Regler 1 Istwert zu Regler 2 Istwert (ab MFU_SE_7_3 n.u.)

[31..27] n.u.

[26..16] Multiplikator (11 Bit) für den Differenzbilder Regler 2

[15..11] n.u.

[10..0] Multiplikator (11 Bit) für den Differenzbilder Regler 1

Name	FSP062_LocalSetValue
Adresse	0x3E_H/62_D/0x3632_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x000000 _H

Repräsentiert den lokalen (Hand) StromSOLLwert.

[23..0] Vorzeichenbehafteter 20 Bit Wert (0xFFFF bis 0x7FFF), wird automatisch dem Sollwert_A zugewiesen, wenn der Schalter Local/Remote auf Local geschaltet ist.

Name	FSP063_MPS
Adresse	0x3F_H/63_D/0x3346_{ASCII}
Tiefe	7 Byte / 56 Bit
I/O	lesen/schreiben
Reset	0x00_0000_0000_0000 _H

With this FSP it is possible to configure the MPS (Machine Protection System) comparator used to generate the MPS_OUT signal sent via LEMO.

[55..53] n.u.

[52] CtrlSel_1L_2H: it selects one of the two available PI controller outputs as value to compare in input to the hysteresis comparator. =1 => controller2 output selected; =0=> controller1 output selected.

[51..49] n.u.

[48] MPS_outSel_DIL_DITH: it selects between the hysteresis comparator output (=0) and the OR combination of the same signal with the tripline (=1) as value of MPS_OUT signal.

[47..24] Histeresys comparator OFF threshold.(20 Bit)

[23..0] Histeresys comparator ON threshold (20 Bit)

Name	FSP064_USI_{HS}_Multiplexer
Adresse	0x40_H/64_D/0x3634_{ASCII}
Tiefe	13 Byte / 103 Bit
I/O	lesen / schreiben
Reset	0x00_0000_0000_0000_0000_0000_0000 _H

Zur Nutzung der USI HighSpeed Transferkanäle darf an eine USI nur maximal ein Modul angeschlossen sein und dieses muss das USI HighSpeed Protokoll unterstützen.

Jede USI unterstützt genau einen USI HighSpeed-Transferkanal in eine Datenrichtung. D.h. ein HighSpeed-Transferkanal von der MFU zum angeschlossene Module mit HighSpeed Unterstützung und ein HighSpeed-Transferkanal vom angeschlossenen Modul mit HighSpeed-Unterstützung zur MFU.

Die Übertragung eines vollständigen Signals benötigt bei maximaler USI Bitrate von 20MBit 6us.

Mit diesem FSP ist eine Vielzahl von Multiplexer parametrierbar, die ein umfangreiches „routen“ von USI HighSpeed Transferkanal Daten von verschiedenen Quellen an verschiedene Ziele ermöglicht.

Die Betrachtung ist dabei wie folgt:

GEHEND[TO] die HighSpeed Daten gehen von der MFU an ein Modul

MFU interne Signale lassen sich hiermit auf USI HighSpeed-Transferkanäle legen. Es können Signale ausgewählt werden, welche via HighSpeed Transferkanal von der MFU über die gewählte USI an das daran angeschlossene Modul mit HighSpeed-Unterstützung übertragen werden.

KOMMEND[FROM] die HighSpeed Daten kommen von einem Modul zur MFU

Hiermit wird definiert von welchem Modul mit HighSpeed-Unterstützung welches Signal in die MFU zurück gelangt. Die Signale dieser USI HighSpeed-Transferkanäle müssen zuvor innerhalb des sendenden Moduls auf den USI HighSpeed-Transferkanal gelegt werden. Jedes Modul hat ggf. standardisierte Daten die im HighSpeed-Transferkanal übertragen werden. Der ADC wird z.B. immer ADC Werte im USI HighSpeed-Transferkanal senden, das ICM immer Netzgeräte-Status-Informationen. Sofern nicht alle Bits des USI HighSpeed-Transferkanals belegt sind ist es möglich wahlweise auf freie Bits zusätzliche Datensignale eines Moduls mittels eines Multiplexers zu legen. Diese zurück gesendeten Daten werden MFU intern wiederum auf bestimmte Signalwege aufgeschaltet, die für eine einwandfreie Funktion des ACU Gesamtsystems notwendig sind.

Kommende [FROM] Signale werden dabei teilweise noch über einen weiteren nachgeschalteten Multiplexer geführt um deren Wertigkeiten und Inhalte weiter zu verfeinern.

[103..100] ControllerX_PreSelectionMux_2 KOMMEND[FROM]

[99..96] ControllerX_PreSelectionMux_1 KOMMEND[FROM]

[95..92] intScope_Chx_PreSelectionMux_2 KOMMEND[FROM]

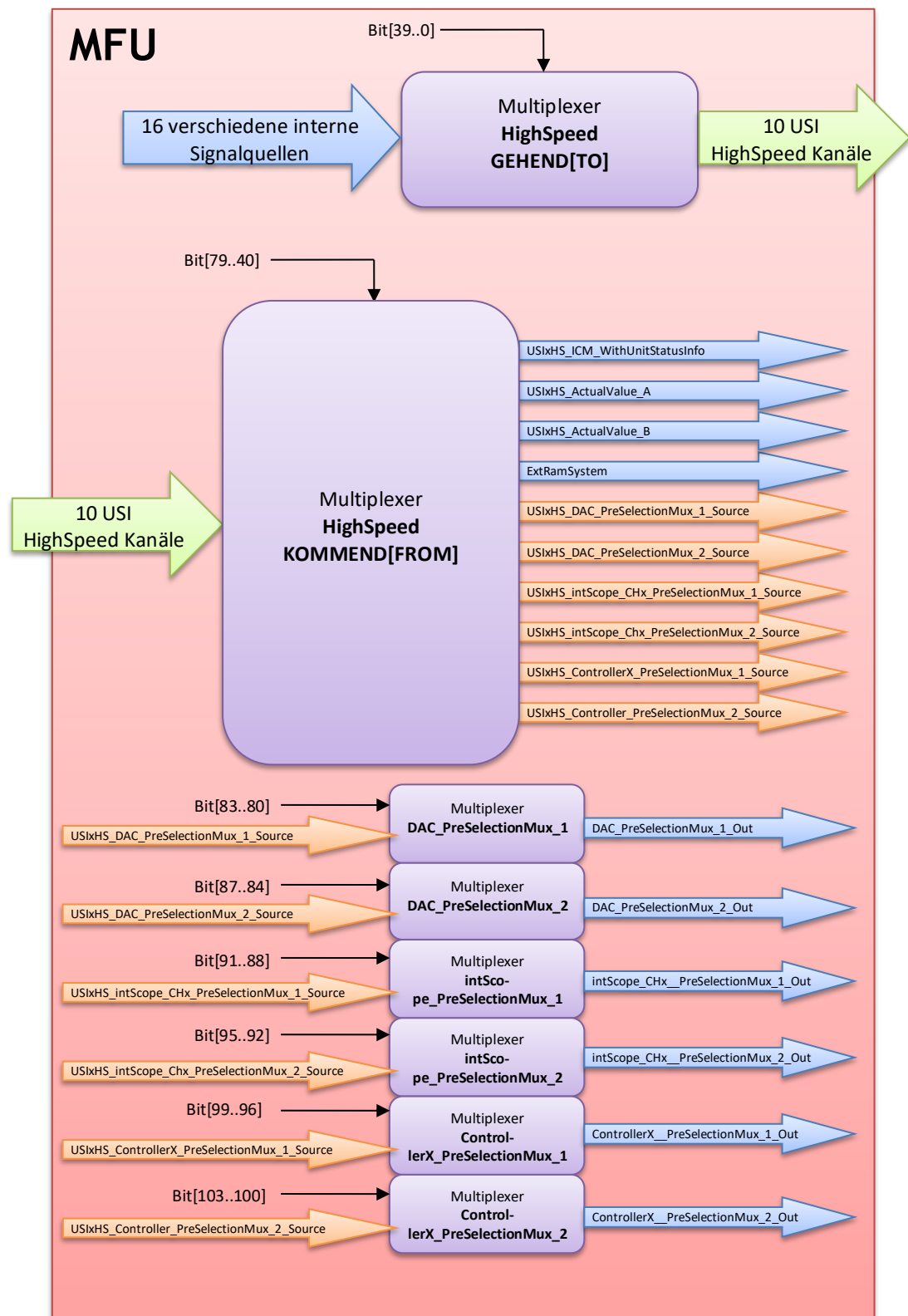
[91..88] intScope_Chx_PreSelectionMux_1 KOMMEND[FROM]

[87..84] DAC_PreSelectionMux_2 KOMMEND[FROM]

[83..80] DAC_PreSelectionMux_1 KOMMEND[FROM]

[79..40] Multiplexer HighSpeed Daten KOMMEND[FROM]

[39..0] Multiplexer HighSpeed GEHEND[TO]



PreSelectionMux ermöglichen beliebige USI HighSpeed Kanäle von den jeweiligen Quellen zum Ziel zu leiten und dabei eine Vorauswahl zu treffen, welche Bit des empfangenen Wertes dabei am Ziel wirklich ankommen sollen.

[103..100] ControllerX_PreSelectionMux_2

Über 4 Bits werden die Daten am Eingang dieses Multiplexers einem Ausgang (ControllerX_PreSelectionMux_2_Out[19..0]) zugewiesen.

[3..0]	Daten am Eingang
--------	------------------

0	n.u
1	USIHS_ControllerX_PreSelectionMux_2_Source[31..12]
2	USIHS_ControllerX_PreSelectionMux_2_Source[31..18],GND_BUS[5..0]
3	USIHS_ControllerX_PreSelectionMux_2_Source[17..4],GND_BUS[5..0]
4	GND_BUS[19..0]
5	GND_BUS[19..0]
6	GND_BUS[19..0]
7	GND_BUS[19..0]
8	GND_BUS[19..0]
9	GND_BUS[19..0]
A	GND_BUS[19..0]
B	GND_BUS[19..0]
C	GND_BUS[19..0]
D	GND_BUS[19..0]
E	GND_BUS[19..0]
F	GND_BUS[19..0]

[99..96] ControllerX_PreSelectionMux_1

Über 4 Bits werden die Daten am Eingang dieses Multiplexers einem Ausgang (ControllerX_PreSelectionMux_1_Out[19..0]) zugewiesen.

[3..0]	Daten am Eingang
0	n.u.
1	USIHS_ControllerX_PreSelectionMux_1_Source[31..12]
2	USIHS_ControllerX_PreSelectionMux_1_Source[31..18],GND_BUS[5..0]
3	USIHS_ControllerX_PreSelectionMux_1_Source[17..4],GND_BUS[5..0]
4	GND_BUS[19..0]
5	GND_BUS[19..0]
6	GND_BUS[19..0]
7	GND_BUS[19..0]
8	GND_BUS[19..0]
9	GND_BUS[19..0]
A	GND_BUS[19..0]
B	GND_BUS[19..0]
C	GND_BUS[19..0]
D	GND_BUS[19..0]
E	GND_BUS[19..0]
F	GND_BUS[19..0]

[95..92] intScope_Chx_PreSelectionMux_2

Über 4 Bits werden die Daten am Eingang dieses Multiplexers einem Ausgang (intScope_Chx_PreSelectionMux_2_Out[19..0]) zugewiesen.

[3..0]	Daten am Eingang
0	n.u.
1	USIHS_intScope_Chx_PreSelectionMux_2_Source[31..12]
2	USIHS_intScope_Chx_PreSelectionMux_2_Source[31..18],GND_BUS[5..0]
3	USIHS_intScope_Chx_PreSelectionMux_2_Source[17..4],GND_BUS[5..0]

4	CurrenComparator_MultipliedDeviation[19..0]
5	GND_BUS[19..0]
6	GND_BUS[19..0]
7	GND_BUS[19..0]
8	GND_BUS[19..0]
9	GND_BUS[19..0]
A	GND_BUS[19..0]
B	GND_BUS[19..0]
C	GND_BUS[19..0]
D	GND_BUS[19..0]
E	GND_BUS[19..0]
F	GND_BUS[19..0]

[91..88] intScope_Chx_PreSelectionMux_1

Über 4 Bits werden die Daten am Eingang dieses Multiplexers einem Ausgang (intScope_Chx_PreSelectionMux_1_Out[19..0]) zugewiesen.

[3..0]	Daten am Eingang
0	n.u.
1	USlXHS_intScope_Chx_PreSelectionMux_1_Source[31..12]
2	USlXHS_intScope_Chx_PreSelectionMux_1_Source[31..18],GND_BUS[5..0]
3	USlXHS_intScope_Chx_PreSelectionMux_1_Source[17..4],GND_BUS[5..0]
4	CurrenComparator_MultipliedDeviation[19..0]
5	GND_BUS[19..0]
6	GND_BUS[19..0]
7	GND_BUS[19..0]
8	GND_BUS[19..0]
9	GND_BUS[19..0]
A	GND_BUS[19..0]
B	GND_BUS[19..0]
C	GND_BUS[19..0]
D	GND_BUS[19..0]
E	GND_BUS[19..0]
F	GND_BUS[19..0]

[87..84] DAC_PreSelectionMux_2

Über 4 Bits werden die Daten am Eingang dieses Multiplexers einem Ausgang (DAC_PreSelectionMux_2_Out[19..0]) zugewiesen.

[3..0]	Daten am Eingang
0	n.u.
1	USlXHS_DAC_PreSelectionMux_2_Source[31..12]
2	USlXHS_DAC_PreSelectionMux_2_Source[31..18],GND_BUS[5..0]
3	USlXHS_DAC_PreSelectionMux_2_Source[17..4],GND_BUS[5..0]
4	CurrenComparator_MultipliedDeviation[19..0]
5	GND_BUS[19..0]
6	GND_BUS[19..0]
7	GND_BUS[19..0]

8	GND_BUS[19..0]
9	GND_BUS[19..0]
A	GND_BUS[19..0]
B	GND_BUS[19..0]
C	GND_BUS[19..0]
D	GND_BUS[19..0]
E	GND_BUS[19..0]
F	GND_BUS[19..0]

[83..80] DAC_PreSelectionMux_1

Über 4 Bits werden die Daten am Eingang dieses Multiplexers einem Ausgang (DAC_PreSelectionMux_1_Out[19..0]) zugewiesen.

[3..0]	Daten am Eingang
0	n.u.
1	USI _x HS_DAC_PreSelectionMux_1_Source[31..12]
2	USI _x HS_DAC_PreSelectionMux_1_Source[31..18], GND_BUS[5..0]
3	USI _x HS_DAC_PreSelectionMux_1_Source[17..4], GND_BUS[5..0]
4	CurrentComparator_MultipliedDeviation[19..0]
5	GND_BUS[19..0]
6	GND_BUS[19..0]
7	GND_BUS[19..0]
8	GND_BUS[19..0]
9	GND_BUS[19..0]
A	GND_BUS[19..0]
B	GND_BUS[19..0]
C	GND_BUS[19..0]
D	GND_BUS[19..0]
E	GND_BUS[19..0]
F	GND_BUS[19..0]

[79..40] Mux_16to10_DataFROMHighSpeedPorts

Dieser Multiplexer hat 16 Eingänge, die in Abhängigkeit der Bitnibbel 10 Ausgängen zugeordnet werden. Die Daten an diesen Ausgängen entsprechen den empfangenen HighSpeedDaten der zugehörigen USI. Z.B. Eingang 2 ist der HighSpeed Empfangskanal von USI 2.

- [79..76] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 10 gelegt wird
- [75..72] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 9 gelegt wird
- [71..68] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 8 gelegt wird
- [67..64] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 7 gelegt wird
- [63..60] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 6 gelegt wird
- [59..56] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 5 gelegt wird
- [55..52] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 4 gelegt wird
- [51..48] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 3 gelegt wird
- [47..44] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 2 gelegt wird
- [43..40] legt fest, welcher HighSpeedPort_x_RcvdData an den Ausgang 1 gelegt wird

Nachfolgend sind die Belegungen der Multiplexereingänge aufgeführt

DataIn	Daten am Eingang
1	HighSpeedPort_1_RcvdData [31..0]
2	HighSpeedPort_2_RcvdData [31..0]
3	HighSpeedPort_3_RcvdData [31..0]
4	HighSpeedPort_4_RcvdData [31..0]
5	HighSpeedPort_5_RcvdData [31..0]
6	HighSpeedPort_6_RcvdData [31..0]
7	HighSpeedPort_7_RcvdData [31..0]
8	HighSpeedPort_8_RcvdData [31..0]
9	HighSpeedPort_9_RcvdData [31..0]
10	HighSpeedPort_10_RcvdData [31..0]

Die Ausgänge erwarten folgende Informationen

DataOut	Daten am Ausgang
1	USIHS_ICM_WithUnitStatusInfo[31..0]
2	USIHS_ActualValue_A[31..0]
3	USIHS_ActualValue_B[31..0]
4	ExtRamSystem[31..0]
5	USIHS_DAC_PreSelectionMux_1_Source[31..0]
6	USIHS_DAC_PreSelectionMux_2_Source[31..0]
7	USIHS_intScope_Chx_PreSelectionMux_1_Source[31..0]
8	USIHS_intScope_Chx_PreSelectionMux_2_Source[31..0]
9	USIHS_ControllerX_PreSelectionMux_1_Source[31..0]
10	USIHS_ControllerX_PreSelectionMux_2_Source[31..0]

An den Bits[79..40] werden die Eingänge (DataIn) für die jeweiligen Ausgänge (DataOut) des Multiplexers gewählt.

Nachfolgend ist aufgeführt welche Bitkombination den jeweiligen Eingang auf welchen Ausgang legt.

Bitkombination	Zuweisung (Daten am Ausgang)
0x0	Eingang 1[31..0]
0x1	Eingang 2[31..0]
0x2	Eingang 3[31..0]
0x3	Eingang 4[31..0]
0x4	Eingang 5[31..0]
0x5	Eingang 6[31..0]
0x6	Eingang 7[31..0]
0x7	Eingang 8[31..0]
0x8	Eingang 9[31..0]
0x9	Eingang 10[31..0]
0xA..0xF	Ausgang ist 0

Beispiel:

Die Bitkombination (Ausgang n) $\rightarrow [(n*4)-1..(n*4)]$ legt fest welcher Eingang (HighSpeedPort_x_RcvdData [31..0]) auf den Ausgang gelegt wird.

Sollen die Daten von Eingang 2 an Ausgang 2 müssen die Bit 47.44 auf 1 stehen

$$[(2 * 4)-1 .. (2 * 4) - 4] = [47..44]$$

Sollen die Daten von Eingang 9 an Ausgang 6 müssen die Bit 63..60 auf 8 stehen

$$[(6 * 4) - 1 \dots (6 * 4) - 4] = [63..60]$$

[39..0] Mux_16to10_DataTOHighSpeedPorts

Dieser Multiplexer hat 16 Eingänge, die in Abhängigkeit der Bitnibbel 10 Ausgängen zugeordnet werden. Die Daten an diesen Ausgängen entsprechen den zu sendenden HighSpeedDaten der zugehörigen USI. Z.B. Ausgang 2 ist der HighSpeed Sendekanal von USI 2.

[39..36] legt fest, welcher Eingang an HighSpeedPort_10_TransferData gelegt wird

[35..32] legt fest, welcher Eingang an HighSpeedPort_9_TransferData gelegt wird

[31..28] legt fest, welcher Eingang an HighSpeedPort_8_TransferData gelegt wird

[27..24] legt fest, welcher Eingang an HighSpeedPort_7_TransferData gelegt wird

[23..20] legt fest, welcher Eingang an HighSpeedPort_6_TransferData gelegt wird

[19..16] legt fest, welcher Eingang an HighSpeedPort_5_TransferData gelegt wird

[15..12] legt fest, welcher Eingang an HighSpeedPort_4_TransferData gelegt wird

[11..8] legt fest, welcher Eingang an HighSpeedPort_3_TransferData gelegt wird

[7..4] legt fest, welcher Eingang an HighSpeedPort_2_TransferData gelegt wird

[3..0] legt fest, welcher Eingang an HighSpeedPort_1_TransferData gelegt wird

Nachfolgend sind die Belegungen der Multiplexereingänge aufgeführt

Dataln	Daten am Eingang
1	[31..5] GND_BUS[26..0] [4] IGBT_V5_Active [3..0] Command[3..0]
2	[31..12] ERLOutValue[19..0] [11..5] GND_BUS[6..0] [4] IGBT_V5_Active [3..0] Command[3..0]
3	[31..12] Controller1_SetValueMuxOut[19..0] [11..5] GND_BUS[6..0] [4] IGBT_V5_Active, [3..0] Command[3..0]
4	[31..12] Controller2_SetValueMuxOut[19..0] [11..5] GND_BUS[6..0] [4] IGBT_V5_Active [3..0] Command[3..0]
5	[31..13] GND_BUS[31..13] [12..9] HighSpeedPort_3_Out[3..0] [8..4] MUX_DesignatedLoad[4..0] [3..0] Command[3..0]
6	[31..12] SetValue_A_MuxOut[19..0] [11..5] GND_BUS[6..0] [4] IGBT_V5_Active [3..0] Command[3..0]
7	[31..12] SetValue_B_MuxOut[19..0] [11..5] GND_BUS[6..0] [4] IGBT_V5_Active [3..1] Command[3..0]
8	[31..0] USIxHS_ICM_WithUnitStatusInfo[31..0]
9	[31..18] Controller1_SetValueDeviation[19..6] [17..4] Controller1_ActValMuxOut[19..6] [3..0] Command[3..0]

10	[31..18] <i>SlopeLimiterOut_C1</i> [19..6] [17..4] <i>Controller1_ActValMuxOut</i> [19..6] [3..0] <i>Command</i> [3..0]
11	[31..18] <i>ERLOutValue</i> [19..6] [17..5] <i>Controller1_SetValueDeviation</i> [19..7] [4] <i>IGBT_V5_Active</i> , [3..0] <i>Command</i> [3..0]
12	[31..18] <i>SlopeLimiterOut_C2</i> [19..6] [17..4] <i>Controller2_ActValMuxOut</i> [19..6] [3..0] <i>Command</i> [3..0]
13	[31..18] <i>Controller1_ActValMuxOut</i> [19..6] [17..4] <i>Controller2_ActValMuxOut</i> [19..6] [3..0] <i>Command</i> [3..0]
14	[31..8] <i>FSP20_ActualValue_A</i> [23..0] [7..5] <i>GND_BUS</i> [2..0] [4] <i>IGBT_V5_Active</i> [3..0] <i>Command</i> [3..0]
15	n.u.
16	n.u.

Nachfolgend ist aufgeführt welche Bitkombination den jeweiligen Eingang auf welchen Ausgang legt.

Bitkombination	Zuweisung (Daten am Ausgang)
0x0	Eingang 1[31..0]
0x1	Eingang 2[31..0]
0x2	Eingang 3[31..0]
0x3	Eingang 4[31..0]
0x4	Eingang 5[31..0]
0x5	Eingang 6[31..0]
0x6	Eingang 7[31..0]
0x7	Eingang 8[31..0]
0x8	Eingang 9[31..0]
0x9	Eingang10[31..0]
0xA	Eingang11[31..0]
0xB	Eingang12[31..0]
0xC	Eingang13[31..0]
0xD	Eingang14[31..0]
0xE	Eingang15[31..0]
0xF	Eingang16[31..0]

Beispiel:

Die Bitkombination (USI n) $\rightarrow [(n*4)-1..(n-4)]$ legen fest welcher Eingang auf den Ausgang (HighSpeedPort_x_TransferData) gelegt wird.

Sollen die Daten von Eingang 2 an HighSpeedPort_2_TransferData (USI2) müssen die Bit 7..4 auf 1 stehen

$$[(2 * 4)-1 .. (2 * 4) -4] = [7..4]$$

Sollen die Daten von Eingang 9 an HighSpeedPort_6_TransferData (USI6) müssen die Bit 23..20 auf 8 stehen

$$[(6 * 4)-1 .. (6 * 4) -4] = [23..20]$$

Name	FSP065_FrontLemoMultiplexer
Adresse	0x41_H/65_D/0x3431_{ASCII}
Tiefe	2 Byte / 16 Bit
I/O	lesen / schreiben
Reset	0x17_17 _H

Repräsentiert die Einstellungen der Ausgangsmultiplexer für die Front-Lemo-Ausgangsbuchsen.

[15..13] n.u.

[12..8] Quellenwahl für Front LEMO Ausgang 2 (X2) (5 Bit)

[4..0]	Ausgang des Multiplexers
0x00	intScopeTriggerEventDetected
0x01	intScopeTimeBasePulse
0x02	ControllerStatusBit[0] (GND)
0x03	ControllerStatusBit[1] (GND)
0x04	ControllerStatusBit[2] (GND)
0x05	ControllerStatusBit[3] (GND)
0x06	ControllerStatusBit[4] (GND)
0x07	ControllerStatusBit[5] (GND)
0x08	ControllerStatusBit[6] (GND)
0x09	ControllerStatusBit[7] (ACU_Controller_Enable)
0x0A	ControllerStatusBit[8] (CurrentEqual)
0x0B	ControllerStatusBit[9] (IGBT_V5_Active)
0x0C	ControllerStatusBit[10] (Controller2_IGBT_V5_Active)
0x0D	ControllerStatusBit[11] (Controller1_IGBT_V5_Active)
0x0E	ControllerStatusBit[12] (PWM_FDRIVE_Comparator_MuxOut)
0x0F	ControllerStatusBit[13] (Controller2_PWM_FDRIVE_Comparator)
0x10	ControllerStatusBit[14] (Controller1_PWM_FDRIVE_Comparator)
0x11	ControllerStatusBit[15] (Degaussing_Active)
0x12	ControllerStatusBit[16] (Controller2_P2_Part_Active)
0x13	ControllerStatusBit[17] (Controller1_P2_Part_Active)
0x14	ControllerStatusBit[18] (Controller2_I_Part_Disable)
0x15	ControllerStatusBit[19] (Controller1_I_Part_Disable)
0x16	1 (VCC)
0x17	0 (GND)
0x18	MPS
0x19	n.u.
0x1A	n.u.
0x1B	n.u.
0x1C	n.u.
0x1D	n.u.
0x1E	n.u.
0x1F	n.u.

[7..5] n.u.

[4..0] Quellenwahl für Front LEMO Ausgang 1 (X1) (5 Bit)

[4..0]	Ausgang des Multiplexers
0x00	intScopeTriggerEventDetected

0x01	intScopeTimeBasePulse
0x02	ControllerStatusBit[0] (GND)
0x03	ControllerStatusBit[1] (GND)
0x04	ControllerStatusBit[2] (GND)
0x05	ControllerStatusBit[3] (GND)
0x06	ControllerStatusBit[4] (GND)
0x07	ControllerStatusBit[5] (GND)
0x08	ControllerStatusBit[6] (GND)
0x09	ControllerStatusBit[7] (ACU_Controller_Enable)
0x0A	ControllerStatusBit[8] (CurrentEqual)
0x0B	ControllerStatusBit[9] (IGBT_V5_Active)
0x0C	ControllerStatusBit[10] (Controller2_IGBT_V5_Active)
0x0D	ControllerStatusBit[11] (Controller1_IGBT_V5_Active)
0x0E	ControllerStatusBit[12] (PWM_FDRIVE_Comparator_MuxOut)
0x0F	ControllerStatusBit[13] (Controller2_PWM_FDRIVE_Comparator)
0x10	ControllerStatusBit[14] (Controller1_PWM_FDRIVE_Comparator)
0x11	ControllerStatusBit[15] (Degaussing_Active)
0x12	ControllerStatusBit[16] (Controller2_P2_Part_Active)
0x13	ControllerStatusBit[17] (Controller1_P2_Part_Active)
0x14	ControllerStatusBit[18] (Controller2_I_Part_Disable)
0x15	ControllerStatusBit[19] (Controller1_I_Part_Disable)
0x16	1 (VCC)
0x17	0 (GND)
0x18	MPS
0x19	n.u.
0x1A	n.u.
0x1B	n.u.
0x1C	n.u.
0x1D	n.u.
0x1E	n.u.
0x1F	n.u.

Name	FSP066_ModuleInterlockInfosForSCU
Adresse	0x42_H/66_D/0x3432_{ASCII}
Tiefe	128 Byte / 1024 Bit
I/O	lesen / schreiben
Reset	0xFFFF.... _H

INFO: Ab MFU FW 7.5.0 entfällt der FSP066. Die Übertragen von Interlockinfos an die SCU/die IFK ist nun RAM organisiert.

Kommuniziert die von den Modulen an den USIs abgefragten Interlocks nach außen zur weiteren Verarbeitung über die Interfacekarte/SCU. Es werden die aktuell anstehenden Interlocks übermittelt.

Die verwendete Tiefe dieses FSP kann an dieser Stelle nicht genau definiert werden. Der sicherste Weg die Größe dieses FSP herauszufinden ist es direkt zu lesen oder per MDS die Tiefe zu ermitteln.

Dies aus dem folgendem Grund:

Das Füllen dieses FSP übernimmt der NIOS, dabei kommt es darauf an, wie viele Module an den USIs gefunden werden und wie viele Interlocks jedes Modul zur Verfügung stellt.

Beispiel:

USI1: 1 Modul mit 10 Interlocks

USI2: 1 Modul mit 24 Interlocks

USI4: 1 Modul mit 4 Interlocks

USI6: 1 Modul mit 24 Interlocks

In der Summe werden also mindestens $10+24+4+24=62$ Interlockbits benötigt. Interlocks werden aber immer in Speichertiefen mit vielfachen von 8 verarbeitet, d.h. für das Modul an USI1 werden 16 Bit und das Modul an USI4 8 Bit verwendet. In diesem Beispiel müsste FSP66 also mindestens 72 Bit tief sein. Unbelegte Interlockbits bleiben dabei frei.

Das Füllen des FSP66 geschieht wie folgt:

Beginnend von Bit[0] werden die Interlockinformationen der einzelnen USIs und deren Module aufsteigend zum MSB eingetragen.

[71..48] 24 Interlockbits USI6, Modul 1

[47..44] frei

[43..40] 4 Interlockbits USI 4, Modul 1

[39..16] 24 Interlockbits USI2, Modul 1

[15..10] frei

[9..0] 10 Interlockbits USI1, Modul 1

Um eine gewisse Sicherheit die Tiefe des FSP betreffend zu haben ist dieses FSP i.d.R. ausreichend groß zu definieren, dass auch ein System im Vollausbau in der Lage ist alle Interlockbits in diesem FSP zu sichern.

Name	FSP067_Defined_USI
Adresse	0x43_H/67_D/0x3433_{ASCII}
Tiefe	2 Byte / 16 Bit
I/O	lesen / schreiben
Reset	0x0000H

This FSP is used to enable the High Speed data received/sent from/to MFU. It is written during the power converter setup and it contains the information of which USI is defined in the configuration file.

[15..10] Not used.

[9] Enable High Speed data for USI nr10 (active high)

[8] Enable High Speed data for USI nr9 (active high)

[7] Enable High Speed data for USI nr8 (active high)

[6] Enable High Speed data for USI nr7 (active high)

[5] Enable High Speed data for USI nr6 (active high)

[4] Enable High Speed data for USI nr5 (active high)

[3] Enable High Speed data for USI nr4 (active high)

[2] Enable High Speed data for USI nr3 (active high)

[1] Enable High Speed data for USI nr2 (active high)

[0] Enable High Speed data for USI nr1 (active high)

Name	FSP068_ButtonAndLEMOInStatus
Adresse	0x44_H/68_D/0x3434_{ASCII}
Tiefe	1 Byte / 8 Bit
I/O	Lesen
Reset	0x(siehe Beschreibung) _H

Stellt die Status der Tasten, Schalter und LEMO Eingänge zur Verfügung.

- [7] Status LEMO_IN[2] (X4)
- [6] Status LEMO_IN[1] (X3)
- [5] n.u., immer ,0'
- [4] n.u., immer ,0'
- [3] Status Schalter REMOTE/LOCAL, ,1' = REMOTE
- [2] Status RESET Taste, ,1' = gedrückt
- [1] Status OFF Taste, ,1' = gedrückt
- [0] Status ON Taste, ,1' = gedrückt

Name	FSP069_ExternalTriplinesStatus
Adresse	0x45_H/69_D/0x3435_{ASCII}
Tiefe	4 Byte / 32 Bit
I/O	Lesen
Reset	0x(siehe Beschreibung) _H

Stellt die Status der externen Reißleinen zur Verfügung.

[31..26] n.u., immer ,0‘

[25..16] ExtTripLinesPulled_n, tatsächlicher Zustand der externen Reißleinen. Eine Reißleine gilt als geschlossen, sofern das zugehörige Bit gesetzt (,1‘) ist.

[15..10] n.u., immer ,0‘

[9..0] ExtTripLinesPulled_nMem, gespeicherter Zustand der externen Reißleinen. Eine Reißleine gilt als geschlossen, sofern das zugehörige Bit gesetzt (,1‘) ist. Wird eine Reißleine geöffnet wird sofort der Zustand aller Reißleinen gespeichert und hier ausgegeben. Die Reißleine die zuerst ausgelöst wurde ist ,0‘, alle übrigen Bits blieben unverändert. Erst wenn alle Reißleinen wieder geschlossen sind, wird die Ausgabe von , ExtTripLinesPulled_nMem‘ sich wieder auf ,0x11_1111_1111‘ ändern.

Name	FSP070_Controller_1_2_InputSourceSelectionMultiplexer
Adresse	0x46_H/70_D/0x3436_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x00_0_0_0_0 _H

Repräsentiert die Einstellungen der PI Regler **1** und **2** Eingangsmultiplexer für dessen Soll- und Istwertquelle.

[23..20] n.u.

[19] '1' Ausgangswert des Regler **2** Istwertquellen-Multiplexers wird invertiert

[18] '1' Ausgangswert des Regler **2** Sollwertquellen-Multiplexers wird invertiert

[17] '1' Ausgangswert des Regler **1** Istwertquellen-Multiplexers wird invertiert

[16] '1' Ausgangswert des Regler **1** Sollwertquellen-Multiplexers wird invertiert

[15..12] Regler **2** Quellenwahl für das Istwertquellen-Multiplexer-Ausgangssignal (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A , bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B , bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C , FSP032_SetValue_C
0x4	Sollwert D , FSP033_SetValue_D
0x5	Istwert A , FSP020_ActualValue_A
0x6	Istwert B , FSP021_ActualValue_B
0x7	Ausgang interner Funktionsgenerator
0x8	Summand 1 für den Addierer, bestimmt durch “FSP090_Adder_1_2_SourceSelectionMultiplexer[16][3..0]“
0x9	Controller 1 _SetValueDeviation, analog FSP076_Controller_1_SetValueDeviation
0xA	Controller X _PreSelectionMux_1_Out, bestimmt durch FSP064_USIxHS_Multiplexer[99..96]
0xB	Controller X _PreSelectionMux_2_Out, bestimmt durch FSP064_USIxHS_Multiplexer[103..100]
0xC	GND_BUS
0xD	GND_BUS
0xE	GND_BUS
0xF	GND_BUS

[11..8] Regler **2** Quellenwahl für Sollwertquellen-Multiplexer-Ausgangssignal (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A , bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B , bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C , FSP032_SetValue_C
0x4	Sollwert D , FSP033_SetValue_D
0x5	Istwert A , FSP020_ActualValue_A
0x6	Istwert B , FSP021_ActualValue_B
0x7	Ausgang interner Funktionsgenerator
0x8	Summand 1 für den Addierer, bestimmt durch

	“FSP090_Adder_1_2_SourceSelectionMultiplexer[16][3..0]“
0x9	Controller1_SetValueDeviation, analog FSP076_Controller_1_SetValueDeviation
0xA	ControllerX_PreSelectionMux_1_Out, bestimmt durch FSP064_USIxHS_Multiplexer[99..96]
0xB	ControllerX_PreSelectionMux_2_Out, bestimmt durch FSP064_USIxHS_Multiplexer[103..100]
0xC	GND_BUS
0xD	GND_BUS
0xE	GND_BUS
0xF	GND_BUS

[7..4] Regler 1 Quellenwahl für das Istwertquellen-Multiplexer-Ausgangssignal (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A, bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B, bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C, FSP032_SetValue_C
0x4	Sollwert D, FSP033_SetValue_D
0x5	Istwert A, FSP020_ActualValue_A
0x6	Istwert B, FSP021_ActualValue_B
0x7	Ausgang interner Funktionsgenerator
0x8	ControllerX_PreSelectionMux_1_Out, bestimmt durch FSP064_USIxHS_Multiplexer[99..96]
0x9	ControllerX_PreSelectionMux_2_Out, bestimmt durch FSP064_USIxHS_Multiplexer[103..100]
0xA	GND_BUS
0xB	GND_BUS
0xC	GND_BUS
0xD	GND_BUS
0xE	GND_BUS
0xF	GND_BUS

[3..0] Regler 1 Quellenwahl für Sollwertquellen-Multiplexer-Ausgangssignal (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A, bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B, bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C, FSP032_SetValue_C
0x4	Sollwert D, FSP033_SetValue_D
0x5	Istwert A, FSP020_ActualValue_A
0x6	Istwert B, FSP021_ActualValue_B
0x7	Ausgang interner Funktionsgenerator
0x8	ControllerX_PreSelectionMux_1_Out, bestimmt durch FSP064_USIxHS_Multiplexer[99..96]
0x9	ControllerX_PreSelectionMux_2_Out, bestimmt durch FSP064_USIxHS_Multiplexer[103..100]
0xA	GND_BUS
0xB	GND_BUS
0xC	GND_BUS

0xD	GND_BUS
0xE	GND_BUS
0xF	GND_BUS

Name	FSP071_Controller_1_SetValue
Adresse	0x47_H/71_D/0x3731_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann der Sollwert des PI Regler **1** ausgelesen werden. Der Sollwert wird dabei über den Multiplexer „FSP070_Controller_1_2_InputSourceSelectionMultiplexer“ ausgewählt.

Das Äquivalent für Regler **1** ist der „FSP071_Controller_1_SetValue“.

[23..0] Vorzeichenbehafteter 20 Bit Sollwert des PI Regler **1**

Name	FSP072_Controller_1_ActualValue
Adresse	0x48_H/72_D/0x3438_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann der Istwert des PI Regler **1** ausgelesen werden. Der Istwert wird dabei über den Multiplexer „FSP070_Controller_1_2_InputSourceSelectionMultiplexer“ ausgewählt.

Das Äquivalent für Regler **2** ist der „FSP082_Controller_2_ActualValue“.

[23..0] vorzeichenbehafteter 20 Bit Istwert des PI Regler **1**

Name	FSP073_Controller_1_Limits
Adresse	0x49_H/73_D/0x3439_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Über diesen FSP können die Bereichsgrenzen des PI Regler **1** festgelegt werden.

Das Äquivalent für Regler **2** ist der „FSP083_Controller_2_Limits“.

[47..24] Controller_1_MaxVal (20 Bit), repräsentiert den oberen (maximalen) Grenzwert des PI Regler **1**

[23..0] Controller_1_MinVal (20 Bit), repräsentiert den unteren (minimalen) Grenzwert des PI Regler **1**

Name	FSP074_Controller_1_PI_Settings
Adresse	0x4A_H/60_D/0x3441_{ASCII}
Tiefe	13 Byte / 104 Bit
I/O	lesen / schreiben
Reset	0x00_00000000_00000000_00000000 _H

Über diesen FSP können die I und P1, P2 Anteile des Regler **1** gesetzt werden.

Das Äquivalent für Regler **2** ist der „FSP084_Controller_2_PI_Settings“.

[103..97] n.u.

[96] Controller_1_PI_Control - Wenn ,1' wird der I-Anteil des Reglers um den Faktor 1000 verlangsamt.

[95..64] Controller_1_I_Part (32 Bit), repräsentiert den I Anteil des PI Regler **1**.

[63..32] Controller_1_P2_Part (32 Bit), repräsentiert den 2. P Anteil des PI Regler **1**.

[63..18] Vorkommastelle (Integer)

[17..0] Nachkommastelle

[31..0] Controller_1_P1_Part (32 Bit), repräsentiert den 1. P Anteil des PI Regler **1**.

[31..20] Vorkommastelle (Integer)

[19..0] Nachkommastelle

Name	FSP075_Controller_1_I_Part_ComparatorLimits
Adresse	0x4B_H/75_D/0x3442_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Repräsentiert die Bereichsgrenzen in denen der I Anteil bei der Regelung berücksichtigt werden soll.

Das Äquivalent für Regler **2** ist der „FSP085_Controller2_I_Part_ComparatorLimits“.

[47..24] Controller_1_I_Part_ComparatorOFFThreshold (20 Bit)

[23..0] Controller_1_I_Part_ComparatorONThreshold (20 Bit)

Name	FSP076_Controller_1_SetValueDeviation
Adresse	0x4C_H/76_D/0x3443_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann die Regeldifferenz (Delta I) des PI Regler **1** gelesen werden. Die Regeldifferenz wird dabei im Differenzbildner des Regler **1** aus dem Minuend SetValue (Ausgang: „FSP070_Controller_1_2_InputSourceSelectionMultiplexer“, bzw. „FSP071_Controller_1_SetValue“) und dem Subtrahend ActualValue (Ausgang: „FSP070_Controller_1_2_InputSourceSelectionMultiplexer“, bzw. „FSP072_Controller_1_ActualValue“).

Das Äquivalent für Regler **2** ist der „FSP086_Controller_2_SetValueDeviation“.

[23..0] SetValueDeviation (20 Bit)

Name	FSP077_Controller_1_PI_Output
Adresse	0x4D_H/77_D/0x3444_{ASCII}
Tiefe	9 Byte / 72 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP können der PI-Anteil, I-Anteil und P-Anteil des PI Regler **1** gelesen werden

Das Äquivalent für Regler **2** ist der „FSP087_Controller_2_PI_Output“.

[71..48] Controller_1_P_Part_Output(20 Bit)

[47..24] Controller_1_I_Part_Output (20 Bit)

[23..0] Controller_1_PI_Output (20 Bit)

Name	FSP078_Controller_1_P2_Part_ComparatorLimits
Adresse	0x4E_H/78_D/0x3445_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Repräsentiert die Bereichsgrenzen in denen der P2 Anteil bei der Regelung des Regler **1** berücksichtigt werden soll.

Das Äquivalent für Regler **2** ist der „FSP088_Controller_2_P2_Part_ComparatorLimits“.

[47..24] Controller_1_P2_Part_ComparatorOFFThreshold (20 Bit)

[23..0] Controller_1_P2_Part_ComparatorONThreshold (20 Bit)

Name	FSP079_Controller_1_SlopeLimiterOutput
Adresse	0x4F_H/79_D/0x34465_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Liefert den Ausgangswert des Steilheitsbegrenzers für Regler **1**

Das Äquivalent für Regler **2** ist der „FSP089_Controller_2_SlopeLimiterOutput“.

[23..0] SlopeLimiterOutput Regler **1** (20 Bit)

Name	FSP081_Controller_2_SetValue
Adresse	0x51_H/81_D/0x3531_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann der Sollwert des PI Regler **2** ausgelesen werden. Der Sollwert wird dabei über den Multiplexer „FSP070_Controller_1_2_InputSourceSelectionMultiplexer“ ausgewählt.

Das Äquivalent für Regler **2** ist der „FSP071_Controller_1_SetValue“.

[23..0] vorzeichenbehafteter 20 Bit Sollwert des PI Regler **2**

Name	FSP082_Controller_2_ActualValue
Adresse	0x52_H/82_D/0x3532_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann der Istwert des PI Regler **2** ausgelesen werden. Der Istwert wird dabei über den Multiplexer „FSP070_Controller_1_2_InputSourceSelectionMultiplexer“ ausgewählt.

Das Äquivalent für Regler **2** ist der „FSP071_Controller_1_SetValue“.

[23..0] vorzeichenbehafteter 20 Bit Istwert des PI Regler **2**

Name	FSP083_Controller_2_Limits
Adresse	0x53_H/83_D/0x3533_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Über diesen FSP können die Bereichsgrenzen des PI Regler **2** festgelegt werden.

Das Äquivalent für Regler **1** ist der „FSP073_Controller_1_Limits“.

[47..24] Controller2_MaxVal (20 Bit), repräsentiert den oberen (maximalen) Grenzwert des PI Regler **2**

[23..0] Controller2_MinVal (20 Bit), repräsentiert den unteren (minimalen) Grenzwert des PI Regler **2**

Name	FSP084_Controller_2_PI_Settings
Adresse	0x54_H/84_D/0x3534_{ASCII}
Tiefe	13 Byte / 104 Bit
I/O	lesen / schreiben
Reset	0x00_00000000_00000000_00000000 _H

Über diesen FSP können die I und P1, P2 Anteile des Reglers **2** gesetzt werden.

Das Äquivalent für Regler **1** ist der „FSP074_Controller_1_PI_Settings“.

[103..96] Controller_2_PI_Control

[103..97] n.u.

[96] Wenn ,1' wird der I-Anteil des Reglers um den Faktor 1000 verlangsamt.

[95..64] Controller_2_I_Part (32 Bit), repräsentiert den I Anteil des PI Regler **2**.

[63..32] Controller_2_P2_Part (32 Bit), repräsentiert den 2. P Anteil des PI Regler **2**.

[31..0] Controller_2_P1_Part (32 Bit), repräsentiert den 1. P Anteil des PI Regler **2**.

Name	FSP085_Controller2_I_Part_ComparatorLimits
Adresse	0x55_H/85_D/0x3535_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Repräsentiert die Bereichsgrenzen in denen der I Anteil bei der Regelung berücksichtigt werden soll.

Das Äquivalent für Regler 1 ist der „FSP075_Controller_1_I_Part_ComparatorLimits“.

[47..24] Controller_2_I_Part_ComparatorOFFThreshold (20 Bit)

[23..0] Controller_2_I_Part_ComparatorONThreshold (20 Bit)

Name	FSP086_Controller_2_SetValueDeviation
Adresse	0x56_H/86_D/0x3536_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann die Regeldifferenz (Delta I) des PI Regler **2** gelesen werden. Die Regeldifferenz wird dabei im Differenzbildner des Regler **2** aus dem Minuend SetValue (Ausgang: „FSP070_Controller_1_2_InputSourceSelectionMultiplexer“, bzw. „FSP081_Controller_2_SetValue“) und dem Subtrahend ActualValue (Ausgang: „FSP070_Controller_1_2_InputSourceSelectionMultiplexer“, bzw. „FSP082_Controller_2_ActualValue“).

Das Äquivalent für Regler **1** ist der „FSP086_Controller_2_SetValueDeviation“.

[23..0] SetValueDeviation (20 Bit)

Name	FSP087_Controller_2_PI_Output
Adresse	0x57_H/87_D/0x3837_{ASCII}
Tiefe	9 Byte / 72 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP können der PI-Anteil, I-Anteil und P-Anteil des PI Regler **2** gelesen werden

Das Äquivalent für Regler **1** ist der „FSP077_Controller_1_PI_Output“.

[71..48] Controller_2_P_Part_Output(20 Bit)

[47..24] Controller_2_I_Part_Output (20 Bit)

[23..0] Controller_2_PI_Output (20 Bit)

Name	FSP088_Controller_2_P2_Part_ComparatorLimits
Adresse	0x58_H/88_D/0x3838_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Repräsentiert die Bereichsgrenzen in denen der P2 Anteil bei der Regelung des Regler **2** berücksichtigt werden soll.

Das Äquivalent für Regler **1** ist der „FSP078_Controller_1_P2_Part_ComparatorLimits“.

[47..24] Controller_2_P2_Part_ComparatorOFFThreshold (20 Bit)

[23..0] Controller_2_P2_Part_ComparatorONThreshold (20 Bit)

Name	FSP089_Controller_2_SlopeLimiterOutput
Adresse	0x59_H/89_D/0x3839_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Liefert den Ausgangswert des Steilheitsbegrenzers für Regler **2**

Das Äquivalent für Regler **1** ist der „FSP079_Controller_1_SlopeLimiterOutput“.

[23..0] SlopeLimiterOutput Regler **2** (20 Bit)

Name	FSP090_Adder_1_2_SourceSelectionMultiplexer
Adresse	0x5A_H/90_D/0x3541_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x00_0_0_0_0 _H

Repräsentiert die Einstellungen der Addierer Eingangsmultiplexer für dessen Summanden 1 und 2.

[23..20] n.u.

[19] '1' Ausgangswert des Multiplexers [Adder_2_Summand_2] wird invertiert

[18] '1' Ausgangswert des Multiplexers [Adder_2_Summand_1] wird invertiert

[17] '1' Ausgangswert des Multiplexers [Adder_1_Summand_2] wird invertiert

[16] '1' Ausgangswert des Multiplexers [Adder_1_Summand_1] wird invertiert

[15..12] Quellenwahl für Multiplexer Ausgangssignal [Adder_1_Summand_2] (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A, bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B, bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C, FSP032_SetValue_C
0x4	Sollwert D, FSP033_SetValue_D
0x5	Controller1_P_Part_Output
0x6	Controller1_I_Part_Output
0x7	Controller1_PI_Part_Output
0x8	Controller2_P_Part_Output
0x9	Controller2_I_Part_Output
0xA	Controller2_PI_Part_Output
0xB	ControllerX_PreSelectionMux_1_Out, bestimmt durch FSP064_USIxHS_Multiplexer[99..96]
0xC	ControllerX_PreSelectionMux_2_Out, bestimmt durch FSP064_USIxHS_Multiplexer[103..100]
0xD	GND_BUS
0xE	GND_BUS
0xF	GND_BUS

[11..8] Quellenwahl für Multiplexer Ausgangssignal [Adder_2_Summand_1] (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A, bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B, bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C, FSP032_SetValue_C
0x4	Sollwert D, FSP033_SetValue_D
0x5	Controller1_P_Part_Output
0x6	Controller1_I_Part_Output
0x7	Controller1_PI_Part_Output
0x8	Controller2_P_Part_Output
0x9	Controller2_I_Part_Output
0xA	Controller2_PI_Part_Output
0xB	ControllerX_PreSelectionMux_1_Out, bestimmt

	durchFSP064_USIcHS_Multiplexer[99..96]
0xC	ControllerX_PreSelectionMux_2_Out, bestimmt durchFSP064_USIcHS_Multiplexer[103..100]
0xD	GND_BUS
0xE	GND_BUS
0xF	GND_BUS

[7..4] Quellenwahl für Multiplexer Ausgangssignal [Adder_1_Summand_2] (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A, bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B, bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C, FSP032_SetValue_C
0x4	Sollwert D, FSP033_SetValue_D
0x5	Istwert A, FSP020_ActualValue_A
0x6	Istwert B, FSP021_ActualValue_B
0x7	Ausgang interner Funktionsgenerator
0x8	Controller1_SetValueDeviation, analog FSP076_Controller_1_SetValueDeviation
0x9	Controller1_P_Part_Output
0xA	Controller1_I_Part_Output
0xB	Controller1_PI_Part_Output
0xC	Controller2_SetValueDeviation, analog FSP086_Controller_2_SetValueDeviation
0xD	Controller2_P_Part_Output
0xE	Controller2_I_Part_Output
0xF	Controller2_PI_Part_Output

[3..0] Quellenwahl für Multiplexer Ausgangssignal [Adder_1_Summand_1] (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A, bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B, bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C, FSP032_SetValue_C
0x4	Sollwert D, FSP033_SetValue_D
0x5	Istwert A, FSP020_ActualValue_A
0x6	Istwert B, FSP021_ActualValue_B
0x7	Ausgang interner Funktionsgenerator
0x8	Controller1_SetValueDeviation, analog FSP076_Controller_1_SetValueDeviation
0x9	Controller1_P_Part_Output
0xA	Controller1_I_Part_Output
0xB	Controller1_PI_Part_Output
0xC	Switched_Contr._PI_Output, in Abhängigkeit der Degausingfunktion und FSP095_ComparatorControl[5]
0xD	SlopeLimiterOutput_C1[19..0]
0xE	GND_BUS
0xF	GND_BUS

Name	FSP091_Adder_1_2_Limits
Adresse	0x5B_H/91_D/0x3542_{ASCII}
Tiefe	12 Byte / 96 Bit
I/O	lesen / schreiben
Reset	0x000000_000000_000000_000000 _H

Über diesen FSP werden die Bereichsgrenzen des Addierers festgelegt.

- [95..72] Adder_2_MaxVal (20 Bit), repräsentiert den oberen (maximalen) Grenzwert des Addierer 2.
- [71..48] Adder_2_MinVal (20 Bit), repräsentiert den unteren (minimalen) Grenzwert des Addierer 2.
- [47..24] Adder_1_MaxVal (20 Bit), repräsentiert den oberen (maximalen) Grenzwert des Addierer 1.
- [23..0] Adder_1_MinVal (20 Bit), repräsentiert den unteren (minimalen) Grenzwert des Addierer 1.

Name	FSP092_Adder_1_SumOut
Adresse	0x5C_H/92_D/0x3543_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann die Summe des Addierwerkes 1 gelesen werden. Die Summanden werden dabei über „FSP090_Adder_1_2_SourceSelectionMultiplexer“ die Bereichsgrenzen über „FSP091_Adder_1_2_Limits“ festgelegt.

[23..0] Adder_1_SumOut (20 Bit)

Name	FSP093_CorrFactorPI_Limits
Adresse	0x5D_H/93_D/0x3544_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Über diesen FSP werden die Bereichsgrenzen des CorrectorFactor PI Regler festgelegt.

[47..24] MaxVal (20 Bit), repräsentiert den oberen (maximalen) Grenzwert.

[23..0] MinVal (20 Bit), repräsentiert den unteren (minimalen) Grenzwert.

Name	FSP094_CorrFactorPI_kP
Adresse	0x5E_H/94_D/0x3545_{ASCII}
Tiefe	4 Byte / 32 Bit
I/O	lesen / schreiben
Reset	0x00000000 _H

Über diesen FSP können die P Anteile des CorrectionFactor Reglers gesetzt werden.

[31..0] CorrFactorPI_kP (32 Bit), repräsentiert den P Anteil des CorrFactorPI.

Name	FSP095_ComparatorControl
Adresse	0x5F_H/95_D/0x3546_{ASCII}
Tiefe	2 Byte / 16 Bit
I/O	lesen / schreiben
Reset	0x0000 _H

Repräsentiert verschiedene Komparator-Kontrollzustände in der MFU.

- [15..13] n.u.
- [12] ,1' aktiviert den Komparator zum Aktivieren/Deaktivieren folgender Funktion, sofern die Komparatorgrenzen „FSP103_PWM_FDrive2_ComparatorLimits“ durch die Regelabweichung des Regler 2 über-, bzw. unterschritten werden und Degaussing (Entmagnetisierung) aktiviert ist:

Es wird bei Komparatorauslösung anstelle des Ergebnis der Addierstufe „FSP092_Adder_1_SumOut“ der Wert „FSP105_IGBT_AlternateSet“ als Steuerwert der Rückspeisungsbegrenzung verwendet.
- [11] ,1' aktiviert den Komparator zum Aktivieren/Deaktivieren folgender Funktion, sofern die Komparatorgrenzen „FSP102_PWM_FDrive1_ComparatorLimits“ durch die Regelabweichung des Regler 1 über-, bzw. unterschritten werden und Degaussing (Entmagnetisierung) nicht aktiviert ist:

Es wird bei Komparatorauslösung anstelle des Ergebnis der Addierstufe „FSP092_Adder_1_SumOut“ der Wert „FSP105_IGBT_AlternateSet“ als Steuerwert Rückspeisungsbegrenzung verwendet.
- [10] ,1' aktiviert die Rückspeisungsbegrenzung
- [9] n.u.
- [8] ,1' aktiviert den Komparator zum Aktivieren/Deaktivieren des P2-Anteils beim Regler 2, sofern dieser die Komparatorgrenzen „FSP088_Controller_2_P2_Part_ComparatorLimits“ durch die Regelabweichung des Regler 2 über-, bzw. unterschreitet.
- [7] ,1' aktiviert den Komparator zum Aktivieren/Deaktivieren des I-Anteils beim Regler 2, sofern dieser die Komparatorgrenzen „FSP085_Controller2_I_Part_ComparatorLimits“ durch die Regelabweichung des Regler 2 über-, bzw. unterschreitet.
- [6] n.u.
- [5] ,1' aktiviert die Degaussing- (Entmagnetisierungs)funktion und den Komparator für deren Durchführung, sofern dieser die Komparatorgrenzen „FSP101_Degauss_ComparatorLimit“ durch den Sollwert des Regler 1 über-, bzw. unterschreitet.
- [4] n.u.
- [3] ,1' aktiviert den Komparator zum Aktivieren/Deaktivieren folgender Funktion, sofern die Komparatorgrenzen „FSP100_V5_ComparatorLimits“ durch die Regelabweichung des Regler 1 über-, bzw. unterschritten werden und Degaussing (Entmagnetisierung) nicht aktiviert ist:

,1' aktiviert den Komparator zum Aktivieren/Deaktivieren folgender Funktion, sofern die Komparatorgrenzen „FSP100_V5_ComparatorLimits“ durch die Regelabweichung des Regler 2 über-, bzw. unterschritten werden und Degaussing (Entmagnetisierung) aktiviert ist:

Es wird bei Komparatorauslösung IGBT V5 aktiviert
- [2] n.u.
- [1] ,1' aktiviert den Komparator zum Aktivieren/Deaktivieren des P2-Anteils beim Regler 1, sofern die Reglerabweichung des Regler 1 „FSP078_Controller_1_P2_Part_ComparatorLimits“ über-, bzw. unterschreitet.

- [0] ,1' aktiviert den Komparator zum Aktivieren/Deaktivieren des I-Anteils beim Regler 1, sofern dieser die Komparatorgrenzen „FSP075_Controller_1_I_Part_ComparatorLimits“ über-, bzw. unterschreitet.

Name	FSP97_SelVal2CompP2Comp
Adresse	0x61_H/97_D/0x3631_{ASCII}
Tiefe	1 Byte / 8 Bit
I/O	lesen / schreiben
Reset	0x00 _H

Über dieses FSP kann für die Komparatoren zum aktivieren/deaktivieren des P2 Teils der PI-Regler 1 und 2 der zu vergleichende Wert gewählt werden.

Im normalen Modus ist dies immer die errechnete Reglabweichung zwischen Soll- und Istwert des Reglers zugehörigen Differenzbilders.

Im ExpertMode können andere Vergleichswerte ausgewählt werden.

[7..4] Legt den Vergleichswert für den P2 Komparator des Regler 1 fest.

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_SetValueDeviation[19..0]
0x2	Controller2_SetValueDeviation[19..0]
0x3	SlopeLimiterOutput[19..0] SlopeLimiterOutputC1[19..0] [MFU SE]
0x4	Controller2_SetValueMuxOut[19..0]
0x5	Controller1_ActValueMuxOut[19..0]
0x6	Controller2_ActValueMuxOut[19..0]
0x7	n.u. SlopeLimiterOutput_C2[19..0] [MFU SE]
0x8	n.u.
0x9	n.u.
0xA	n.u.
0xB	n.u.
0xC	n.u.
0xD	n.u.
0xE	n.u.
0xF	n.u.

[3..0] Legt den Vergleichswert für den P2 Komparator des Regler 1 fest.

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_SetValueDeviation[19..0]
0x2	Controller2_SetValueDeviation[19..0]
0x3	SlopeLimiterOutput[19..0] SlopeLimiterOutputC1[19..0] [MFU SE]
0x4	Controller2_SetValueMuxOut[19..0]
0x5	Controller1_ActValueMuxOut[19..0]
0x6	Controller2_ActValueMuxOut[19..0]
0x7	n.u. SlopeLimiterOutput_C2[19..0] [MFU SE]
0x8	n.u.
0x9	n.u.
0xA	n.u.

0xB	n.u.
0xC	n.u.
0xD	n.u.
0xE	n.u.
0xF	n.u.

Name	FSP098_Selectable_klP1
Adresse	0x62_H/98_D/0x3632_{ASCII}
Tiefe	32 Byte / 256 Bit
I/O	lesen / schreiben
Reset	0x0000000000000000_0000000000000000_0000000000000000_0000000000 000000 _H

This FSP contains the 4 of 5 selectable coefficients (kl, kP1) for the controller number one. The first coefficients set is stored in the FSP

- [255..192] Fifth Coefficients set 64=32+32 bits.
 - [63..32] kl_5 coefficient
 - [31..0] kP1_5 coefficient
- [191..128] Fourth Coefficients set 64=32+32 bits.
 - [63..32] kl_4 coefficient
 - [31..0] kP1_4 coefficient
- [127..64] Third Coefficients set 64=32+32 bits.
 - [63..32] kl_3 coefficient
 - [31..0] kP1_3 coefficient
- [63..0] Second Coefficients set 64=32+32 bits.
 - [63..32] kl_2 coefficient
 - [31..0] kP1_2 coefficient

Name	FSP099_Selectable_klK1Thresholds
Adresse	0x63_H/99_D/0x3633_{ASCII}
Tiefe	21 Byte / 168 Bit
I/O	lesen / schreiben
Reset	0x00_00000_00000_00000_00000_00000_00000_00000_00000 _H

This FSP contains the thresholds (on and off) used to generate the coefficients multiplexer selector. The SelectorSignal is compared with these thresholds and a decoded signal is generated for the purpose above described. The FSP contains also the Enable bit used to turn on/off the coefficientSelector module. In off mode, the first coefficients set is always selected (module back compatible)

[167..161] n.u.

[160] Enable:

0=> The first coefficients set are selected to the output (trans-parent mode).

1=> The output coefficients take one of the 5 available inputs based on the SelectorSignal value.

[159..140] Fourth Off Threshold value (20b)

[139..120] Third Off Threshold value (20b)

[119..100] Second Off Threshold value (20b)

[99..80] First Off Threshold value (20b)

[79..60] Fourth On Threshold value (20b)

[59..40] Third On Threshold value (20b)

[39..20] Second On Threshold value (20b)

[19..0] First On Threshold value (20b)

Name	FSP100_V5_ComparatorLimits
Adresse	0x64_H/100_D/0x3634_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Legt die Grenzwerte des Komparators zur Aktivierung/Deaktivierung von IGBT V5 fest.

[47..24] V5_ComparatorOFFThreshold (20 Bit)

[23..0] V5_ComparatorONThreshold (20 Bit)

Name	FSP101_Degauss_ComparatorLimit
Adresse	0x65_H/101_D/0x3635_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0xFFF6A4_00095B _H

Repräsentiert die Schwelle des Komparators für das Entmagnetisieren (Degaussen)

[47..24] Degauss_ComparatorOFFThreshold (20 Bit)

[23..0] Degauss_ComparatorONThreshold (20 Bit)

Name	FSP102_PWM_FDrive1_ComparatorLimits
Adresse	0x66_H/102_D/0x3636_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Legt die Grenzwerte des Komparators zur Aktivierung/Deaktivierung des alternativen Steuerwerts der Rückspeisungsbegrenzung fest, sofern Degaussen (Entmagnetisieren) nicht aktiv ist.

[47..24] PWM_FDRIVE1_ComparatorOFFThreshold (20 Bit)

[23..0] PWM_FDRIVE1_ComparatorONThreshold (20 Bit)

Name	FSP103_PWM_FDrive2_ComparatorLimits
Adresse	0x67_H/103_D/0x3637_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Legt die Grenzwerte des Komparators zur Aktivierung/Deaktivierung des alternativen Steuerwerts der Rückspeisungsbegrenzung fest, sofern Degaussen (Entmagnetisieren) aktiv ist.

[47..24] PWM_FDRIVE2_ComparatorOFFThreshold (20 Bit)

[23..0] PWM_FDRIVE2_ComparatorONThreshold (20 Bit)

Name	FSP104_CorrFactor_Selector
Adresse	0x68_H/104_D/0x3638_{ASCII}
Tiefe	1 Byte / 8 Bit
I/O	lesen/schreiben
Reset	Reset:0x00 _H

It selects the input value used as correction factor.

[7..4] n.u.

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert A, bestimmt durch „FSP013_PeripheralConfig[0]“
0x2	Sollwert B, bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	Sollwert C, FSP032_SetValue_C
0x4	Sollwert D, FSP033_SetValue_D
0x5	Istwert A, FSP020_ActualValue_A
0x6	Istwert B, FSP021_ActualValue_B
0x7	Ausgang interner Funktionsgenerator
0x8	ControllerX_PreSelectionMux_1_Out, bestimmt durch FSP064_USIxHS_Multiplexer[99..96]
0x9	ControllerX_PreSelectionMux_2_Out, bestimmt durch FSP064_USIxHS_Multiplexer[103..100]
0xA	GND_BUS
0xB	GND_BUS
0xC	GND_BUS
0xD	GND_BUS

Name	FSP105_IGBT_AlternateSetValue
Adresse	0x69_H/105_D/0x3639_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x000000 _H

Repräsentiert den alternativen Steuerwert für die Rückspeisungsbegrenzung, sofern einer der Komparatoren die Grenzen „FSP102_PWM_FDrive1_ComparatorLimits“, bei nicht aktivem Degaussing (Entmagnetisieren), bzw. „FSP103_PWM_FDrive2_ComparatorLimits“, bei aktivem Degaussing (Entmagnetisieren) unter-, bzw. überschritten hat.

[23..0] IGBT_AlternateSetValue (20 Bit)

Name	FSP106_EnergyRecoverLimitation_CurrentDriveValue
Adresse	0x6A_H/106_D/0x3641_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Funktion

[47..24] ERL_CurrentThresholdValue (20 Bit)

[23..0] ERL_CurrentDriveValue (20 Bit)

Name	FSP107_DCCT_AdjustmentFactors
Adresse	0x6B_H/107_D/0x3642_{ASCII}
Tiefe	8 Byte / 64 Bit
I/O	lesen / schreiben
Reset	0x00000000_00000000 _H

Korrekturwerte (P-Anteile) für den DCCT Abgleich von ActualValue_A (Istwert_A) und ActualValue_B (Istwert_B)

[63..32] DCCTAdjustmentFactor_ActualValue_B (32 Bit)

[31..0] DCCTAdjustmentFactor_ActualValue_A (32 Bit)

Name	FSP108_CorrFactor_AdderLimits
Adresse	0x6C_H/108_D/0x3643_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen / schreiben
Reset	0x000000_000000 _H

Über diesen FSP werden die Bereichsgrenzen des CorrectorFactor Adder festgelegt.

[47..24] MaxVal (20 Bit), repräsentiert den oberen (maximalen) Grenzwert.

[23..0] MinVal (20 Bit), repräsentiert den unteren (minimalen) Grenzwert.

Name	FSP109_CorrectionFactorSignals
Adresse	0x6D_H/109_D/0x3644_{ASCII}
Tiefe	9 Byte / 72 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

It contains the correction factors values to be monitored:

- [71..48] Correction value selected (20b)
- [47..24] Correctionfactor PI output: P part (20b)
- [23..0] Correctionfactor adder output: SetValue+ Correctionfactor PI output: P part (20b)

Name	FSP110_DACxSourceSelectionMultiplexer FSP110_DACx_and_ScopeChannelx_SourceSelectionMultiplexer
Adresse	0x6E_H/110_D/0x3645_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x00_0_1_1_1 _H

Repräsentiert die Einstellungen der DAC Ausgangs- und int. Oszilloskop Eingangsmultiplexer.

Ab MFU FW 7.5.0 gilt:

Beginnend mit MFU FW 7.5.0 werden sowohl die Ausgabequellen der DACs, als auch die Eingangsquellen des internen Oszilloskops über einen einzigen Multiplexer geführt. Dadurch entfällt auch der FSP115_intScopeSourceSelectionMultiplexer. Das interne Oszilloskop hat nur noch 4 Kanäle, die dabei jeweils den nachfolgenden Quellen entsprechen.

[23..20] n.u.

[19] '1' Ausgangswert des Multiplexers [DAC_4_and_ScopeChannel_4_Source]
→ X8 wird invertiert

[18] '1' Ausgangswert des Multiplexers [DAC_3_and_ScopeChannel_3_Source]
→ X7 wird invertiert

[17] '1' Ausgangswert des Multiplexers [DAC_2_and_ScopeChannel_2_Source]
→ X6 wird invertiert

[16] '1' Ausgangswert des Multiplexers [DAC_1_and_ScopeChannel_1_Source]
→ X5 wird invertiert

[15..12] Quellenwahl für Multiplexer Ausgangssignal [DAC_4_and_ScopeChannel_4_Source] → X8

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_SetValueMuxOut[19..0], Sollwert Regler 1, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[16][3..0]“
0x2	Controller2_SetValueMuxOut[19..0], Sollwert Regler 2, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[18][11..8]“
0x3	DAC_PreSelectionMux_1_Out[19..0], bestimmt durch “FSP064_USIxHS_Multiplexer[83..80]”
0x4	DAC_PreSelectionMux_2_Out[19..0], bestimmt durch “FSP064_USIxHS_Multiplexer[87..84]”
0x5	Controller1_P_Part_Ouput[19..0]
0x6	Controller1_I_Part_Ouput[19..0]
0x7	Controller1_PI_Part_Ouput[19..0]
0x8	Controller2_P_Part_Ouput[19..0]
0x9	Controller2_I_Part_Ouput[19..0]
0xA	Controller2_PI_Part_Ouput[19..0]
0xB	Adder_1_SumOut[19..0]
0xC	ERLOutValue, Rückspeisungsbegrenzung[19..0]
0xD	DeterminedOutValue[19..0], festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xE	ControllerStatusBits[19..0] entspricht dem FSP121_ControllerStatusBits LE/SE [19] Controller_1_I_Part_Disable [18] Controller_2_I_Part_Disable [17] Controller_1_P2_Part_Active [16] Controller_2_P2_Part_Active

[15]	Degaussing_Active
[14]	Controller_1_PWM_FDrive_Comparator
[13]	Controller_2_PWM_FDrive_Comparator
[12]	PWM_FDriveComparator_MuxOut
[11]	Controller_1_IGBT_V5_Active
[10]	Controller_2_IGBT_V5_Active
[9]	IGBT_V5_Active
[8]	n.u., immer 0'
[7]	ACU_Controller_Enable
[6]	SlopeLimiterEnable_1
[5]	n.u., immer 0'
[4]	SlopeLimiterEnable_C2
[3..0]	n.u., immer 0'
0xF	n.u., immer 0

[11..8] Quellenwahl für Multiplexer Ausgangssignal [DAC_3_and_ScopeChannel_3_Source] → X7

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_MultipliedDeviation[19..0], Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Reglerdifferenz
0x2	Controller2_MultipliedDeviation[19..0], Ausgang des MFU_DifferenceCalculator, Regler 2, multiplizierte Reglerdifferenz
0x3	DAC_PreSelectionMux_1_Out[19..0], bestimmt durch „FSP064_USIxHS_Multiplexer[83..80]“
0x4	DAC_PreSelectionMux_2_Out[19..0], bestimmt durch „FSP064_USIxHS_Multiplexer[87..84]“
0x5	Controller1_P_Part_Ouput[19..0]
0x6	Controller1_I_Part_Ouput[19..0]
0x7	Controller1_PI_Part_Ouput[19..0]
0x8	Controller2_P_Part_Ouput[19..0]
0x9	Controller2_I_Part_Ouput[19..0]
0xA	Controller2_PI_Part_Ouput[19..0]
0xB	Adder_1_SumOut[19..0]
0xC	ERLOutValue[19..0], Rückspeisungsbegrenzung
0xD	DeterminedOutValue[19..0], festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xE	n.u., immer 0
0xF	n.u., immer 0

[7..4] Quellenwahl für Multiplexer Ausgangssignal [DAC_2_and_ScopeChannel_2_Source] → X6

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_ActValueMuxOut[19..0], Istwert des Regler 1, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[17][7..4]“
0x2	SetValue_B_MultiplexerOutput[19..0], bestimmt durch „FSP013_PeripheralConfig[0]“
0x3	ActualValue_A[19..0], analog „FSP020_ActualValue_A“
0x4	ActualValue_B[19..0], analog „FSP021_ActualValue_B“
0x5	SlopeLimiterOutput_C2[19..0]
0x6	Controller1_MultipliedDeviation[19..0], Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Reglerdifferenz
0x7	Controller2_MultipliedDeviation[19..0], Ausgang des MFU_DifferenceCalculator, Regler 2, multiplizierte Reglerdifferenz

0x8	DAC_PreSelectionMux_1_Out[19..0], bestimmt durch „FSP064_USIcHS_Multiplexer[83..80]“
0x9	DAC_PreSelectionMux_2_Out[19..0], bestimmt durch „FSP064_USIcHS_Multiplexer[87..84]“
0xA	Controller1_P_Part_Ouput[19..0]
0xB	Controller1_I_Part_Ouput[19..0]
0xC	Controller1_PI_Part_Ouput[19..0]
0xD	ERLOutValue[19..0], Rückspeisungsbegrenzung
0xE	DeterminedOutValue[19..0], festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xF	intFuncGenOutput[19..0]

[3..0] Quellenwahl für Multiplexer Ausgangssignal [DAC_1_and_ScopeChannel_1_Source] → X5

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_SetValueMuxOut[19..0], Sollwert Regler 1, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[16][3..0]“
0x2	Controller2_SetValueMuxOut[19..0], Sollwert Regler 2, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[18][11..8], „
0x3	SlopeLimiterOutput_C1[19..0]
0x4	Controller1_MultipliedDeviation[19..0], Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Regeldifferenz
0x5	Controller2_MultipliedDeviation[19..0], Ausgang des MFU_DifferenceCalculator, Regler 2, multiplizierte Regeldifferenz
0x6	intFuncGenOutput[19..0]
0x7	DAC_PreSelectionMux_1_Out[19..0], bestimmt durch „FSP064_USIcHS_Multiplexer[83..80]“
0x8	DAC_PreSelectionMux_2_Out[19..0], bestimmt durch „FSP064_USIcHS_Multiplexer[87..84]“
0x9	Controller1_I_Part_Ouput[19..0]
0xA	Controller1_PI_Part_Ouput[19..0]
0xB	ERLOutValue[19..0], Rückspeisungsbegrenzung
0xC	DeterminedOutValue[19..0], festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xD	n.u., immer 0
0xE	n.u., immer 0
0xF	n.u., immer 0

Bis MFU FW 7.4.x gilt:

[23..20] n.u.

[19] '1' Ausgangswert des Multiplexers [DAC_4_Input] → X8 wird invertiert

[18] '1' Ausgangswert des Multiplexers [DAC_3_Input] → X7 wird invertiert

[17] '1' Ausgangswert des Multiplexers [DAC_2_Input] → X6 wird invertiert

[16] '1' Ausgangswert des Multiplexers [DAC_1_Input] → X5 wird invertiert

[15..12] Quellenwahl für Multiplexer Ausgangssignal [DAC_4_Input] → X8

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_SetValueMuxOut, Sollwert Regler 1, bestimmt durch

	„FSP070_Controller_1_2_InputSourceSelectionMultiplexer[16][3..0]“
0x2	Controller2_SetValueMuxOut, Sollwert Regler 2, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[18][11..8],„
0x3	DAC_PreSelectionMux_1_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[83..80]“
0x4	DAC_PreSelectionMux_2_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[87..84]“
0x5	Controller1_P_Part_Ouput
0x6	Controller1_I_Part_Ouput
0x7	Controller1_PI_Part_Ouput
0x8	Controller2_P_Part_Ouput
0x9	Controller2_I_Part_Ouput
0xA	Controller2_PI_Part_Ouput
0xB	Adder_1_SumOut
0xC	Adder_2_SumOut
0xD	ERLOutValue, Rückspeisungsbegrenzung
0xE	DeterminedOutValue, festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xF	intFuncGenOutput

[11..8] Quellenwahl für Multiplexer Ausgangssignal [DAC_3_Input] → X7

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Regeldifferenz
0x2	Controller2_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 2, multiplizierte Regeldifferenz
0x3	DAC_PreSelectionMux_1_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[83..80]“
0x4	DAC_PreSelectionMux_2_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[87..84]“
0x5	Controller1_P_Part_Ouput
0x6	Controller1_I_Part_Ouput
0x7	Controller1_PI_Part_Ouput
0x8	Controller2_P_Part_Ouput
0x9	Controller2_I_Part_Ouput
0xA	Controller2_PI_Part_Ouput
0xB	Adder_1_SumOut
0xC	Adder_2_SumOut
0xD	ERLOutValue, Rückspeisungsbegrenzung
0xE	DeterminedOutValue, festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xF	intFuncGenOutput

[7..4] Quellenwahl für Multiplexer Ausgangssignal [DAC_2_Input] → X6

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller1_ActValueMuxOut, Istwert des Regler 1, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[17][7..4]“
0x2	SetValue_B_MultiplexerOutput, bestimmt durch „FSP013_PeripheralConfig[0]“

0x3	ActualValue_A, analog „FSP020_ActualValue_A“
0x4	ActualValue_B, analog „FSP021_ActualValue_B“
0x5	SlopeLimiterOutput
0x6	Controller1_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Reglerdifferenz
0x7	Controller2_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 2, multiplizierte Reglerdifferenz
0x8	DAC_PreSelectionMux_1_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[83..80]“
0x9	DAC_PreSelectionMux_2_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[87..84]“
0xA	Controller1_P_Part_Ouput
0xB	Controller1_I_Part_Ouput
0xC	Controller1_PI_Part_Ouput
0xD	ERLOutValue, Rückspeisungsbegrenzung
0xE	DeterminedOutValue, festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xF	intFuncGenOutput

[3..0]

Quellenwahl für Multiplexer Ausgangssignal [DAC_1_Input] → X5

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Sollwert des Regler 1, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[17][3..1]“
0x2	Sollwert des Regler 2, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[19][11..8]“
0x3	Sollwert C, FSP032_SetValue_C
0x4	Sollwert D, FSP033_SetValue_D
0x5	SlopeLimiterOutput
0x6	Controller1_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Reglerdifferenz
0x7	Controller2_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 2, multiplizierte Reglerdifferenz
0x8	intFuncGenOutput, Ausgang interner Funktionsgenerator
0x9	DAC_PreSelectionMux_1_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[83..80]“
0xA	DAC_PreSelectionMux_2_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[87..84]“
0xB	Controller1_P_Part_Output
0xC	Controller1_I_Part_Otuput
0xD	Controller1_PI_Part_Output
0xE	ERLOutValue, Rückspeisungsbegrenzung
0xF	DeterminedOutValue, festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“

Name	FSP111_DACGain_Offset
Adresse	0x6F_H/111_D/0x3646_{ASCII}
Tiefe	24 Byte / 192 Bit
I/O	lesen / schreiben
Reset	0x400000_000000_400000_000000_400000_000000_400000_000000 _H

Repräsentiert den Verstärkungsfaktor und Offsetwert des 4-fachen DAC für die vorne an der MFU vorhandenen. DAC Diagnosebuchsen.

- [191..168] DAC 4 (Lemo X8) Gain, vorzeichenbehafteter 19 Bit Wert
- [167..144] DAC 4 (Lemo X8) Offset, vorzeichenbehafteter 18 Bit Wert
- [143..120] DAC 3 (Lemo X7) Gain, vorzeichenbehafteter 19 Bit Wert
- [119..96] DAC 3 (Lemo X7) Offset, vorzeichenbehafteter 18 Bit Wert
- [95..72] DAC 2 (Lemo X6) Gain, vorzeichenbehafteter 19 Bit Wert
- [71..48] DAC 2 (Lemo X6) Offset, vorzeichenbehafteter 18 Bit Wert
- [47..24] DAC 1 (Lemo X5) Gain, vorzeichenbehafteter 19 Bit Wert
- [23..0] DAC 1 (Lemo X5) Offset, vorzeichenbehafteter 18 Bit Wert

Name	FSP112_extRAMTriggerStatus
Adresse	0x70_H/112_D/0x3730_{ASCII}
Tiefe	1 Byte / 8 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

It is the direct decoding of the Trigger Finite State Machine.

[7..4] n.u., always ,0'

[3..0] FiniteStaeMachine

[3..0]	
0x0	Not allowed state
0x1	Trigger not enabled
0x3	PreFill RAM state
0x2	Searching trigger event
0x6	Trigger found, wait for the complete write action
0x4	Write complete, now it is possible to read back the data.
0x5	Alarm State.

Name	FSP113_AdderStatus
Adresse	0x71_H/113_D/0x3731_{ASCII}
Tiefe	120 Byte / 15 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

This FSP contains the status of adder 1 and adder2:

- [119..96] Adder_2_SumOut(20b): Adder 2 output status
- [95..72] Adder_2_Summand2_Multiplexer(20b): Adder 2 input 2
- [71..48] Adder_2_Summand1_Multiplexer(20b): Adder 2 input 1
- [47..24] Adder_1_Summand2_Multiplexer(20b): Adder 1 input 2
- [23..0] Adder_1_Summand1_Multiplexer(20b): Adder 1 input 1

Name	FSP114_intScopeTFTSettings
Adresse	0x72_H/114_D/0x3732_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen / schreiben
Reset	0x01_210_0 _H

Repräsentiert die Einstellungen für die Anzeige des internen Scopes auf dem TFT Standardbildschirm.

[23..17] n.u.

[16] intScopeTFT_ShowClassicView

Wenn ,1' wird anstelle des internen Scopes auf dem TFT in jedem Fall die klassische grafische Ansicht des Stromistwertes angezeigt.

Wenn ,0' wird im ,Remote'-Betrieb und wenn KEIN PC mit der MFU verbunden ist im Triggerfall das interne Scope auf dem TFT dargestellt. In allen anderen Fällen die klassische grafische Ansicht des Stromistwertes.

[15..4] intScopeTFT_Channel_Selector

[15] n.u.

[14..12] TFT Kanal 3 (grün)

[11] n.u.

[10..8] TFT Kanal 2 (rot)

[7] n.u.

[6..4] TFT Kanal 1 (blau)

mit

Bitfolge	Aufzeichnung
000	Scope Eingang Kanal 1
001	Scope Eingang Kanal 2
010	Scope Eingang Kanal 3
011	Scope Eingang Kanal 4
100	Scope Eingang Kanal 5
101	Scope Eingang Kanal 6
110	Scope Eingang Kanal 7
111	Scope Eingang Kanal 8

Die **rot** markierten Kanäle stehen ab der MFU FW 7.5.0 nicht mehr zur Verfügung.

[3] n.u.

[2..0] intScopeTFT_Y_Sculling

Skaliert die Auflösung der aufgezeichneten Kanäle für die TFT Darstellung. Jeder Scopeeingangskanal hat eine Auflösung von 12 Bit, die intern für die Darstellung auf dem TFT auf 7 Bit reduziert werden. Von den zwölf Bit werden bei bipolaren Geräten alle 12 Bit benutzt, das MSB bildet dabei das Vorzeichen. Bei unipolaren Geräten werden hingegen nur 11 Bit benutzt, das Vorzeichen bleibt unbenutzt. Bei bipolaren Geräten hat dies zur Folge, dass 6 Bit die eigentliche Spannung und 1 Bit das Vorzeichen repräsentieren. Bei unipolaren Geräten wird die Spannung hingegen auf allen 7 Bit abgebildet. Die Skalierung verschiebt nun diese Bitfolge nach rechts. Dies erhöht die Auflösung im kleinen Spannungsbereich, da die oberen Bit ignoriert werden. Bei bipolaren Geräten kann dadurch die Anzeige um den Faktor 25 = 32-fach und bei unipolaren Geräten 24 = 16-fach vergrößert werden.

Bitfolge	Aufzeichnung
000	Bipolar: TFT Speicherbits[Eingangsbits 11(VZ), 10..5] Unipolar: TFT Speicherbits[Eingangsbits 10..4]
001	Bipolar: TFT Speicherbits[Eingangsbits 11(VZ), 9..4] Unipolar: TFT Speicherbits[Eingangsbits 9..3]
010	Bipolar: TFT Speicherbits[Eingangsbits 11(VZ), 8..3] Unipolar: TFT Speicherbits[Eingangsbits 8..2]

011	Bipolar: Unipolar:	TFT Speicherbits[Eingangsbits 11(VZ), 7..2] TFT Speicherbits[Eingangsbits 7..1]
100	Bipolar: Unipolar:	TFT Speicherbits[Eingangsbits 11(VZ), 6..1] TFT Speicherbits[Eingangsbits 6..0]
101..111	Bipolar: Unipolar:	TFT Speicherbits[Eingangsbits 11(VZ), 5..0] TFT Speicherbits[Eingangsbits 6..0]

Name	FSP115_intScopeSourceSelectionMultiplexer
Adresse	0x73_H/115_D/0x3733_{ASCII}
Tiefe	3 Byte / 24 Bit (bis MFU FW 7.4.x)
I/O	lesen / schreiben
Reset	0x00_0_0_0_0 _H

Dieser FSP wird ab MFU FW 7.5.0 nicht weiter unterstützt.

Die int. Oszilloskop-Quellenwahl wird ab MFU FW 7.5.0 über die Multiplexer der DACs ausgeführt. D.h. die Eingangsquellen der DACs sind auch gleichzeitig die Eingangsquellen der 4 int. Oszilloskopkanäle.

D.h. die feste Verdrahtung der Eingangskanäle 1 bis 4 des int. Oszilloskops die bis MFU FW 7.4.x galt ist obsolet.

Bis MFU FW 7.4.x gilt:

Repräsentiert die Einstellungen des internen Oszilloskops Eingangsmultiplexer. Die Kanäle 1 bis 4 des internen Oszilloskops sind fest verdrahtet und nicht änderbar.

Kanal 1: Ausgang Sollwert-Multiplexer, Regler 1

Kanal 2: Ausgang Istwert-Multiplexer, Regler 1

Kanal 3: Ausgang Regelabweichung-Multiplexer, Regler 1

Kanal 4: Ausgang PI Regler, Regler 1

[23..20] n.u.

[19] '1' Ausgangswert des Multiplexers [intScopeInputCH8] wird invertiert

[18] '1' Ausgangswert des Multiplexers [intScopeInputCH7] wird invertiert

[17] '1' Ausgangswert des Multiplexers [intScopeInputCH6] wird invertiert

[16] '1' Ausgangswert des Multiplexers [intScopeInputCH5] wird invertiert

[15..12] Quellenwahl für Multiplexer Ausgangssignal [ScopeInput8] (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller2_SetValueMuxOut, Sollwert Regler 2, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[18][11..8],“
0x2	Sollwert C, analog „FSP032_SetValue_C“
0x3	Sollwert D, analog „FSP033_SetValue_D“
0x4	SlopeLimiterOutput
0x5	Controller2_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Reglerdifferenz
0x6	intFuncGenOutput, Ausgang interner Funktionsgenerator
0x7	intScope_CHx_PreSelectionMux_1_Out, bestimmt durch “FSP064_USIxHS_Multiplexer[]”
0x8	intScope_CHx_PreSelectionMux_2_Out, bestimmt durch “FSP064_USIxHS_Multiplexer[]”
0x9	Controller1_P_Part_Output
0xA	Controller1_I_Part_Output
0xB	ERLOutValue, Rückspeisungsbegrenzung
0xC	DeterminedOutValue, festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xD	Controller2_P_Part_Output
0xE	Controller2_I_Part_Output
0xF	ControllerStatusBits[19..0], dabei handelt es sich um ein binäres Datenwort, bestehend aus 24 Bits, von denen hier 20 aufgeschaltet sind, die verschiedene

	Steuerzustände innerhalb der MFU repräsentieren. Dieses Datenwort sollte binär interpretiert werden.
[19]	Controller_1_I_Part_Disable
[18]	Controller_2_I_Part_Disable
[17]	Controller_1_P2_Part_Active
[16]	Controller_2_P2_Part_Active
[15]	Degaussing_Active
[14]	Controller_1_PWM_FDrive_Comparator
[13]	Controller_2_PWM_FDrive_Comparator
[12]	PWM_FDriveComparator_MuxOut
[11]	Controller_1_IGBT_V5_Active
[10]	Controller_2_IGBT_V5_Active
[9]	IGBT_V5_Active
[8]	CurrentEqual
[7]	ACU_Controller_Enable
[6]	SlopeLimiterEnable
[5]	CurrentEualSwitchEnable
[4..0]	n.u., immer ‚0‘

[11..8] Quellenwahl für Multiplexer Ausgangssignal [ScopeInput7] (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller2_SetValueMuxOut, Sollwert Regler 2, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[18][11..8],„
0x2	Sollwert C, analog „FSP032_SetValue_C“
0x3	Sollwert D, analog „FSP033_SetValue_D“
0x4	SlopeLimiterOutput
0x5	Controller2_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Reglerdifferenz
0x6	intFuncGenOutput, Ausgang interner Funktionsgenerator
0x7	intScope_CHx_PreSelectionMux_1_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[]“
0x8	intScope_CHx_PreSelectionMux_2_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[]“
0x9	Controller1_P_Part_Output
0xA	Controller1_I_Part_Output
0xB	ERLOutValue, Rückspeisungsbegrenzung
0xC	DeterminedOutValue, festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xD	Controller2_P_Part_Output
0xE	Controller2_I_Part_Output
0xF	Controller2_PI_Part_Output

[7..4] Quellenwahl für Multiplexer Ausgangssignal [ScopeInput6] (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller2_SetValueMuxOut, Sollwert Regler 2, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[18][11..8],„
0x2	Sollwert C, analog „FSP032_SetValue_C“
0x3	Sollwert D, analog „FSP033_SetValue_D“
0x4	SlopeLimiterOutput
0x5	Controller2_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Reglerdifferenz

0x6	intFuncGenOutput, Ausgang interner Funktionsgenerator
0x7	intScope_CHx_PreSelectionMux_1_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[.]“
0x8	intScope_CHx_PreSelectionMux_2_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[.]“
0x9	Controller1_P_Part_Output
0xA	Controller1_I_Part_Output
0xB	ERLOutValue, Rückspeisungsbegrenzung
0xC	DeterminedOutValue, festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xD	Controller2_P_Part_Output
0xE	Controller2_I_Part_Output
0xF	Controller2_PI_Part_Output

[3..0] Quellenwahl für Multiplexer Ausgangssignal [ScopeInput5] (4 Bit)

[3..0]	Ausgang des Multiplexers
0x0	0
0x1	Controller2_SetValueMuxOut, Sollwert Regler 2, bestimmt durch „FSP070_Controller_1_2_InputSourceSelectionMultiplexer[18][11..8],“
0x2	Sollwert C, analog „FSP032_SetValue_C“
0x3	Sollwert D, analog „FSP033_SetValue_D“
0x4	SlopeLimiterOutput
0x5	Controller2_MultipliedDeviation, Ausgang des MFU_DifferenceCalculator, Regler 1, multiplizierte Reglerdifferenz
0x6	intFuncGenOutput, Ausgang interner Funktionsgenerator
0x7	intScope_CHx_PreSelectionMux_1_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[.]“
0x8	intScope_CHx_PreSelectionMux_2_Out, bestimmt durch „FSP064_USIxHS_Multiplexer[.]“
0x9	Controller1_P_Part_Output
0xA	Controller1_I_Part_Output
0xB	ERLOutValue, Rückspeisungsbegrenzung
0xC	DeterminedOutValue, festgelegter Sollwert, bestimmt durch „FSP105_IGBT_AlternateSetValue“
0xD	Controller2_P_Part_Output
0xE	Controller2_I_Part_Output
0xF	Controller2_PI_Part_Output

Name	FSP116_intScopeSettings
Adresse	0x74_H/116_D/0x3734_{ASCII}
Tiefe	9 Byte / 72 Bit
I/O	lesen / schreiben
Reset	0x0000_0_0_0000_0000_0_0_0 _H

Repräsentiert die interne Scope Parametrierung

- [71..60] TriggerLevel (12 Bit) Repräsentiert den Triggerlevel des internen Oszilloskops
 [59..56] n.u. (4 Bit), TriggerLevel unterstes Nibble zur Rundung
 [55..52] External RAM extra configuration(4 bits)

[3..0]	
[0]	Debug mode enable. Active high. It selects as data to store in the RAM, the output of a 32 bit counter and as external trigger signal the bit nr [1] of this nibble.
[1]	Debug External trigger signal. It enables also the debug 32 bits counter described above.Active high
[2]	Trigger signal selector: if 0 => internal trigger signal selected; else external trigger signal selected.
[3]	Incoming data sampler signal: if 0=> the incoming data are sampled using the internal Scope timebase pulse signal; else the incoming data are sampled using the HighSpeedPort_2_NewDataStrobe pulse signal.

- [51..48] TimeScale (4 Bit), gibt die Skalierung der Zeitbasis an

[3..0]	Skalierung
0x0	ns, 100ns kleinster Wert bei TimeBase = 0, bzw. 1, Schrittweite 100ns
0x1	µs, 1µs kleinster Wert bei TimeBase = 0, bzw. 1, Schrittweite 1µs
0x2	ms, 1ms kleinster Wert bei TimeBase = 0, bzw. 1, Schrittweite 1ms
0x3	s, 1s kleinster Wert bei TimeBase = 0, bzw. 1, Schrittweite 1s
0x4.. 0xF	n.u., Standardskalierung 100ns wird gewählt

- [47..32] TimeBase (16 Bit), repräsentiert die Zeitbasis (horizontal) des internen Oszilloskop. 'TimeBase' gibt dabei an, wie viele male 'TimeScale' für ein Gesamtbild benötigt wird.
 Bsp.: abhängig von 'TimeScale' gilt für die zeitliche Auflösung bei 500 Bildpunkten:

$$(TimeBase \times TimeScale \times 500) / 10 = Zeit / DIV$$

 TimeBase = 2, TimeScale = 1(us) => 2us => alle 2us wird ein Bildpunkt erfasst.
 Bei 500 Werten bedeutet dies 2us * 500 = 1000us = 1ms für das gesamte Bild.

- [31..25] n.u. (7 Bit)
 [24..16] TriggerOffset (9 Bit), gibt einen Triggeroffset an
 [15..12] n.u. (4 Bit)
 [11..8] TriggerChannel (4 Bit)

[3..0]	Trigger Kanal
0x0	Kanal 1
0x1	Kanal 2
0x2	Kanal 3
0x3	Kanal 4
0x4	Kanal 5
0x5	Kanal 6
0x6	Kanal 7

0x7	Kanal 8
0x8	Externe Triggerquelle über LEMO Buchse X4, LEMO 1 IN
0x9.. 0xF	Interne Triggerquelle über den internen Funktionsgenerator

Die **rot** markierten Kanäle stehen ab der MFU FW 7.5.0 nicht mehr zur Verfügung.

[7..4] Trigger Mode (4 Bit)

[3..0]	Trigger Modus
0x0	Triggerlevel > Wert, Messwertannäherung von unten durch Triggerlevel
0x1	Triggerlevel = Wert
0x2	Triggerlevel < Wert, Messwertannäherung von oben durch Triggerlevel
0x3	positive Triggerflanke über digitalen Eingang LEMO 1 oder den int. Funktionsgenerator
0x4	negative Triggerflanke über digitalen Eingang LEMO 1 oder den int. Funktionsgenerator

[3..0] TriggerConfig (4 Bit)

[3..0]	Beschreibung
0x0	Trigger Konfigurieren, keine Datenaufzeichnung
0x2	auf Triggerereignis warten (Trigger scharf)

Name	FSP117_intScopeTriggerReadOut
Adresse	0x75_H/117_D/0x3735_{ASCII}
Tiefe	4 Byte / 32 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Repräsentiert Informationen über den Triggerprozess des internen Oszilloskops. Die Werte in diesem FSP werden bei einem Triggerereignis aktualisiert.

[31..28] n.u. (immer 0)

[27..24] TriggerStatus[3..0]

[3..0]	Beschreibung
0x0	TriggerIdle (FSP116_intScopeSettings[3..0] FunctionMode auf 0x0, keine Datenaufzeichnung, Trigger kann konfiguriert werden)
0x1	PrefillingRAM (FSP116_intScopeSettings[3..0] FunctionMode wurde auf 0x2 gesetzt, Datenaufzeichnung noch <u>nicht</u> aktiv, warten bis RAM vorgefüllt ist)
0x2	WaitForTriggerEvent (RAM ist vorgefüllt, FSP116_intScopeSettings[3..0] FunctionMode ist auf 0x2, Datenaufzeichnung aktiv, warten auf Triggerereignis)
0x4	TriggerEventOccured (das in FSP116_intScopeSettings[7..4] TriggerMode gewählte Ereignis hat stattgefunden)
0x8	WaitForReadOutData (alle Daten rund um das Triggerereignis sind aufgezeichnet und zur Abholung bereit)

[23..21] n.u. (immer 0)

[20..12] TriggerOffsetEcho[8..0]
entspricht nach dem Triggerereignis dem Wert von FSP116_intScopeSettings[25..16]

[11..9] n.u. (immer 0)

[8..0] TriggerAddress[8..0]
entspricht nach dem Triggerereignis der Trigger-Adresse

Name	FSP118_intScopeDataReadOutAddress
Adresse	0x76_H/118_D/0x3736_{ASCII}
Tiefe	2 Byte / 16 Bit
I/O	lesen / schreiben
Reset	0x0000 _H

Repräsentiert die 9 Bit Adresse des internen Scopespeichers dessen Daten gelesen werden sollen

[15..9] n.u. (7 Bit)

[8..0] ScopeReadOutRAMAddress (9 Bit), repräsentiert die RAM Speicheradresse des internen Oszilloskop die angesprochen werden soll. Das interne Oszilloskop besitzt 8 Kanäle (4 Kanäle ab MFU FW 7.5.0) die in 8 RAM Speicher (4 RAM Speicher ab MFU FW 7.5.0) abgelegt werden können. Dies geschieht zeitgleich, d.h. die jeweils angelegte Adresse adressiert alle 8 RAM Speicher (4 RAM Speicher ab MFU FW 7.5.0) zeitgleich.

Name	FSP119_intScopeDataReadOut
Adresse	0x77_H/119_D/0x3737_{ASCII}
Tiefe	18 Byte / 144 Bit (bis MFU FW 7.4.x) 10 Byte / 80 Bit (ab MFU FW 7.5.0)
I/O	lesen
Reset	0x(siehe Beschreibung)

Repräsentiert die 9 Bit Adresse des internen Scopespeichers, und die darin befindlichen Daten (Scope-RamData_CHx) der Kanäle 1 bis 8 (bis MFU FW 7.4.x), bzw. der Kanäle 1 bis 4 (ab MFU FW 7.5.0).

Die **rot** markierten Daten entfallen ab MFU FW 7.5.0.

[143..140] n.u. (4 Bit, immer 0)

[139..128] ScopeRamData_CH8[11..0] (12 Bit)

[127..124] n.u. (4 Bit, immer 0)

[123..112] ScopeRamData_CH7[11..0] (12 Bit)

[111..108] n.u. (4 Bit, immer 0)

[107..96] ScopeRamData_CH6[11..0] (12 Bit)

[95..92] n.u. (4 Bit, immer 0)

[91..80] ScopeRamData_CH5[11..0] (12 Bit)

[79..76] n.u. (4 Bit, immer 0)

[75..64] ScopeRamData_CH4[11..0] (12 Bit)

[63..60] n.u. (4 Bit, immer 0)

[59..48] ScopeRamData_CH3[11..0] (12 Bit)

[47..44] n.u. (4 Bit, immer 0)

[43..32] ScopeRamData_CH2[11..0] (12 Bit)

[31..28] n.u. (4 Bit, immer 0)

[27..16] ScopeRamData_CH1[11..0] (12 Bit)

[15..9] n.u. (7 Bit, immer 0)

[8..0] ScopeReadOutRAMAddress[8..0], (9 Bit) entspricht dem Eintrag in FSP118_intScopeDataReadOutAddress und dient der Verifizierung ob die Daten auch zu dieser Adresse gehören.

Name	FSP120_intFunctionGenerator
Adresse	0x78_H/120_D/0x3738_{ASCII}
Tiefe	16 Byte / 128 Bit
I/O	lesen / schreiben
Reset	0x0_0_000000_000000_000000_000000_000000 _H

Repräsentiert die Parameter für den internen Funktionsgenerator

[127..124] ModeSelect

[3..0]	ModeSelect
0x0	Funktionsgenerator läuft frei, unabhängig ‚ExtSyncSource‘
0x1	Eingänge ‚ManualDirectionUP_nDown‘ und ‚ManualTriggerCounter‘ sind aktiv
0x2	Rampe aufwärts/abwärts startet nur mit steigender Flanke an ‚ExternalEnableSelect‘
0x3	-
0x4	-
0x5	-
0x6	-
0x7	-
0x8	-
0x9	-
0xA	-
0xB	-
0xC	-
0xD	-
0xE	-
0xF	-

[123..120] ExternalEnableSelect

[3..0]	ExternalEnableSelect
0x0	GND
0x1	GND
0x2	GND
0x3	GND
0x4	GND
0x5	GND
0x6	GND
0x7	GND
0x8	GND
0x9	GND
0xA	GND
0xB	Front LEMO In 2
0xC	Front LEMO In 1
0xD	intScopeTriggerEventDetected
0xE	VCC
0xF	GND

[119..96] RampRiseTime (24 Bit), Wenn X"00_0000" pulst der Funktionsgenerator, andernfalls wird gerammt. RampTime repräsentiert dabei die Pausenzeit zwischen zwei Rampenstützpunkten während der Anstiegs-, bzw. Abstiegszeit der Rampe. Gibt die Zeit der Pausenintervalle zwischen den einzelnen Inkrementen zwischen dem ‚LowermostValueRampPulse‘ und dem ‚TopValueRampPulse‘ an. Das Pausenintervall entspricht „(RampRise * 10ns) + 10 ns“. Beispiel: LowermostValueRampPulse = -32768_D
TopValueRampPulse = 32767_D
RampTime = 1_D
Es liegen 65535 Stützpunkte zwischen dem Minimal und Maximalwert.
Es wird alle „(RampRise * 10ns)“ ein neuer Stützpunkt ausgegeben, d.h. alle 10ns. Eine Rampe dauert also 65535 * (1*10ns)

[95..72]

[95..89] n.u. (7 Bit)

[88..72] LowermostValueDuration (17 Bit), repräsentiert die Zeitdauer die der untere (minimale) Pulswert/Scheitelwert der Rampe (Faltbottom) des internen Funktionsgenerators anstehen soll. Ist Bit [88] gesetzt gilt der Wert von Bit[87..72] in uSekunde. Ist Bit [88] nicht gesetzt gilt der Wert von Bit[87..72] in mSekunden.

[71..48]

[71..65] n.u. (7 Bit)

[64..48] TopValueDuration (17 Bit), repräsentiert die Zeitdauer die der obere (maximale) Pulswert/Scheitelwert der Rampe (Flattp) des internen Funktionsgenerators anstehen soll. Ist Bit [64] gesetzt gilt der Wert von Bit [63..48] in uSekunde. Ist Bit [64] nicht gesetzt gilt der Wert von Bit [63..48] in mSekunden.

[47..24]

[47..28] LowermostValueRampPulse (20 Bit), repräsentiert den unteren (minimalen) Pulswert/Scheitelwert der Rampe des internen Funktionsgenerators. Vorzeichenbehafteter Wert im Bereich zwischen -11..+11 Volt.

[27..24] n.u. (4 Bit)

[23..0]

[23..4] TopValueRampPulse (20 Bit), repräsentiert den oberen (maximalen) Pulswert/Scheitelwert der Rampe des internen Funktionsgenerators. Vorzeichenbehafteter Wert im Bereich zwischen -11..+11 Volt.

[3..0] n.u. (4 Bit)

Name	FSP121_ControllerStatusBits LE/SE
Adresse	0x79_H/121_D/0x3739_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	lesen
Reset	0x01 _H

Hierbei handelt es sich um ein binäres Datenwort, bestehend aus 23 Bits, die verschiedene Steuerzustände innerhalb der MFU repräsentieren.

[23..20]	n.u., immer ,0'
[19]	Controller_1_I_Part_Disable
[18]	Controller_2_I_Part_Disable
[17]	Controller_1_P2_Part_Active
[16]	Controller_2_P2_Part_Active
[15]	Degaussing_Active
[14]	Controller_1_PWM_FDrive_Comparator
[13]	Controller_2_PWM_FDrive_Comparator
[12]	PWM_FDriveComparator_MuxOut
[11]	Controller_1_IGBT_V5_Active
[10]	Controller_2_IGBT_V5_Active
[9]	IGBT_V5_Active
[8]	CurrentEqual / n.u.
[7]	ACU_Controller_Enable
[6]	SlopeLimiterEnable / SlopeLimiterEnable C1
[5]	CurrentEqualSwitchEnable / n.u.
[4]	n.u. / SlopeLimiterEnable C2
[3..0]	n.u., immer ,0'

Name	FSP125_LoadSwitchSelection
Adresse	0x82_H/125_D/0x3832_{ASCII}
Tiefe	1 Byte / 8 Bit
I/O	lesen / schreiben
Reset	0x01 _H

Gibt an, welche Last über das LoadSwitchModul gewählt werden soll, sofern ein LoadSwitchModul vorhanden ist. Der Inhalt des FSP125 muss dabei über den HighSpeed Transferkanal an das LSM gesendet werden. Die Selektion des HighSpeed Transferkanals übernimmt FSP064_USIxHS_Multiplexer.

[7..5] n.u.

[4..0] Lastauswahl (Bitcodiert)

[4..0]	Last
00000	keine Last
00001	Last 1
00010	Last 2
00100	Last 3
01000	Last 4
10000	Last 5

Name	FSP229_SW_HighSpeedStream_Synchronized
Adresse	0xE5_H/229_D/0x4535_{ASCII}
Tiefe	dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

FSP229 basiert auf dem „FSP237_SW_HighSpeedStream“ und gibt ebenfalls eine bestimmte Anzahl von Daten aus, die über einen wählbaren HighSpeed Kanal gesammelt werden.

Bedingung hierbei ist, dass das Zielmodul über diese Funktionalität verfügt. Dazu müssen im Zielmodul die FSP017 und FSP018 vorhanden sein.

Die Unterschiede dabei sind:

- die Datenerfassung ist synchronisiert, d.h. die erfassten Daten werden nicht nach einer voreinstellbaren Zeit gepollt, sondern es wird jeder neue Messwert erfasst
- die Anzahl lesbarer HS Datensätze ist größer 2^{16} (\Rightarrow max. 2^{32}), ist wegen des begrenzten RAM im Modul/der MFU aber nicht nutzbar). Dabei ist die Anzahl lesbarer Messwerte im Modul FSP017 zu finden und wird nicht vom Nutzer gesetzt
- vor dem Lesen wird
 - 1.) der Regler gesperrt
 - 2.) im auszulesenden Modul seitens der MFU eine Initialisierung für den Ausleseprozess gestartet.

Bevor das FSP gelesen werden kann, muss es beschrieben werden mit:

STX PID PID GW MA FSP FSP [USIN_r.] PP PP ETX

Mit:

PID WR
[USIN_r.] USI Nummer (0x01..0x0B)

Im Anschluss wird das FSP gelesen.

Die MFU führt daraufhin die folgenden Aktionen aus:

- sämtliche USIs seitens der MFU werden deselektiert
- es wird geprüft ob die aktuelle MFU Firmware diese Funktion unterstützt
- FSP017 des Zielmoduls lesen. Dort ist die Anzahl der erfassbaren Messwerte zu finden.
- ist FSP017 des Zielmoduls lesbar. die passende Speichergröße in der MFU reservieren
- den Regler sperren
- im Zielmodul FSP018[0] einen Reset der Datenerfassung ausführen
- im Zielmodul FSP018[2] den Trigger zur Datenerfassung aktivieren
- nun wird die Anzahl von Daten laut FSP017 des Zielmoduls, im Zielmodul erfasst und auch dort gespeichert
- FSP017 des Zielmoduls lesen bis Bit[1] gesetzt ist (Messung beendet)
- Zielmodul FSP018[1] setzen, dies ermöglicht das lesen der Messdaten
- die MFU setzt das Kommando „TriggerSometing“ ab, dadurch startet das Zielmodul den Datentransfer der Messwerte über den HighSpeed Kanal. Dazu wird der „normale HighSpeed Stream“ unterbrochen. Stattdessen findet der Messwertdatentransfer statt.
- sind alle Daten übertragen werden diese an die übergeordnete Instanz (Kontrollsystem/Anwender PC) ausgegeben
- der „normale HighSpeed Stream“ wird wieder auf der USI aktiviert

Die Daten sehen dabei wie folgt aus:

STX GW MA FSP FSP
[Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0] ← Datensatz n
[Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0] ← Datensatz n-1
[...]
[Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0] ← Datensatz n-n
PP PP ETX

Name	FSP230_SW_intScopeHeaderReadOut
Adresse	0xE6_H/230_D/0x4536_{ASCII}
Tiefe	6 Byte / 48 Bit
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Der interne SW FSP230 ist ein CPU Software FSP und stellt die TimeScale, TimeBase, TriggerOffsetEcho, und TriggerAddress des internen Oszilloskops über den USB→PC Anschluss zur Verfügung.

Diese Daten werden von PCA (PowerConfigAdvanced) zur Darstellung der internen Oszilloskopdaten benötigt.

FSP230 kann nur gelesen werden. Schreibenanforderungen bekommen eine NACK Antwort mit Fehlercode.

Die Informationen des FSP230 werden dabei aus Inhalten der „FSP116_intScopeSettings“ und „FSP117_intScopeTriggerReadOut“ gewonnen.

Ausgabestring nach der Leseanforderung:

```
STX GW MA -FSP-  -TS--  ----TB-----  ----TOE-----  ---TA---  --PP-  ETX
02  30 30 45 36  ts ts  tt tt tt tt  toe toe toe ta ta ta pp pp  03
```

TS	TimeScale	Eingestellte Zeitskalierung für TimeBase des internen Oszilloskops (2 Byte ASCII = 8 Bit HEX)
TB	Timebase	Eingestellte Zeitbasis des internen Oszilloskops (4 Byte ASCII = 16 Bit HEX)
TOE	TriggerOffsetEcho	Triggeroffset vom Nullpunkt der X-Achse (3 Byte ASCII = 12 Bit HEX)
TA	TriggerAddress	Speicheradresse an der das Triggerereignis stattgefunden hat (3 Byte ASCII = 12 Bit HEX)

Der Speicher des int. Oszilloskops ist als Ringspeicher ausgeführt. D.h. solange kein Triggerereignis erfolgt werden die Daten zyklisch in diesen Ringspeicher eingetragen.

Findet nun ein Triggerereignis statt, wird in TriggerAddress die Adresse gespeichert an der das Ereignis stattgefunden hat, TriggerOffsetEcho gibt die Trigger Offsetverschiebung vom Nullpunkt der X-Achse an.

Name	FSP231_SW_intScopeDataStreamReadOut
Adresse	0xE7_H/231_D/0x4537_{ASCII}
Tiefe	12008 Byte / 96064 Bit (bis MFU FW 7.4.x) 6008 Byte / 48064 Bit (ab MFU FW 7.5.0)
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Der interne FSP231 ist ein CPU Software FSP und stellt die eigentlichen Daten des internen Oszilloskops als Stream über den USB→PC Anschluss zur Verfügung.

Diese Daten werden von PCA (PowerConfigAdvanced) zur Darstellung der internen Oszilloskopdaten benötigt.

FSP231 kann nur gelesen werden. Schreibenanforderungen bekommen eine NACK Antwort mit Fehlercode.

Sollen die Daten beginnend vom Triggeroffset und nicht vom RAM Speicher 0 ausgegeben werden bitte den FSP245_SW_intScopeDataStream benutzen.

Bis MFU FW 7.4.x gilt folgendes:

Ausgabestring nach der Leseanforderung:

```
STX GW MA -FSP- -CH8 (12Bit) CH7 (12Bit) CH7 (12Bit) CH5 (12Bit)
CH4 (12Bit) CH3 (12Bit) CH2 (12Bit) CH1 (12Bit) - -PP- ETX
```

```
02 30 30 45 37 dd dd dd dd dd dd dd dd dd dd dd
dd dd dd dd dd dd dd dd dd dd dd pp pp 03
```

CHx - Daten des Kanals

Insgesamt werden 500 x **8** Kanäle übertragen. D.h. der gelb markierte Bereich wird also insgesamt 500-mal übertragen. Beginnend mit dem Wert 1 für Kanal 8 bis 1, Wert 2 für Kanal 8 bis 1 usw. Die Prüfsumme PP folgt nach Wert 500 für Kanal 8 bis 1 über die gesamten Daten.

Ab MFU FW 7.5.0 gilt folgendes:

Ausgabestring nach der Leseanforderung:

```
STX GW MA -FSP- -CH4 (12Bit) CH3 (12Bit) CH2 (12Bit) CH1 (12Bit) - -PP-
ETX
```

```
02 30 30 45 37 dd dd dd dd dd dd dd dd dd dd pp pp
03
```

CHx - Daten des Kanals

Insgesamt werden 500 x **4** Kanäle übertragen. D.h. der gelb markierte Bereich wird also insgesamt 500-mal übertragen. Beginnend mit dem Wert 1 für Kanal 4 bis 1, Wert 2 für Kanal 4 bis 1 usw. Die Prüfsumme PP folgt nach Wert 500 für Kanal 4 bis 1 über die gesamten Daten.

Name	FSP232_SW_intSystemParameters
Adresse	0xE8_H/232_D/0x4538_{ASCII}
Tiefe	dynamisch
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Der interne FSP232 ist ein CPU Software FSP und stellt die ACU Systemparameter als Stream über den USB→PC Anschluss zur Verfügung.

FSP232 kann nur gelesen werden. Schreibenanforderungen bekommen eine NACK Antwort mit Fehlercode.

Nach einer Leseanforderung werden die in der MFU gespeicherten Systemparameter übertragen. Diesen wird der reguläre USI Header vorangestellt und der Stream mit Prüfsumme und ETX abgeschlossen.

Wichtiger Hinweis:

Die Systemparameter sind als USI konforme Strings im MFU Speicher hinterlegt. d.h. jeder dieser Einzelstrings wird mit einem STX [0x02] eingeleitet und einer Prüfsumme über die Daten dieses Einzelstrings, sowie einem ETX[0x03] abgeschlossen. Die Summe aller Systemparameter wiederum wird ebenfalls als USI konformer String ausgegeben. Dieser wird also ebenfalls durch ein ETX eingeleitet und mit Prüfsumme (diesmal über alle vorherigen Parameterstrings) und ETX abgeschlossen.

□	0	0	E	8	□	W	R	0	0	F	1	0	D	0	1	0	0	7	5	□	□	W	R
02	30	30	45	38	02	57	52	30	30	46	31	30	44	30	31	30	30	37	35	03	02	57	52

Einleitung der FSP Antwort mit STX, Gateway, Moduladresse, FSP Nummer (E8_H = 223_D)

erster Parameterstring

Anfang des zweiten Parameterstring

es folgen ggf. weitere Parameterstrings

0	0	0	0	0	0	0	0	□	4	C	□
30	30	30	30	30	30	30	03	34	43	03	

Rest des letzten Parameterstring

Abschluss der FSP Antwort mit Prüfsumme (über alle Parameterstrings, inkl. deren STX, „WR“, Gateway, Modulnummer, Daten, Prüfsummen und ETX) und ETX

Das Speichern der Systemparameter wird mittels Bit[7] im „FSP242_SW_CPU_Status“ (siehe Seite 123) eingeleitet und auch bestätigt.

Name	FSP233_SW_InterlockTexts
Adresse	0xE9_H/233_D/0x4539_{ASCII}
Tiefe	dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Der interne FSP233 ist ein CPU Software FSP und beinhaltet die für das ACU System in der MFU hinterlegten Interlocktexte.

Weitere Informationen zum beschreiben dieses FSP finden Sie im Dokument "ACU-MFU-FSP233-Interlocktexte programmieren".

Nach einer Leseanforderung werden die in der MFU gespeicherten Interlocktexte übertragen. Diesen wird der reguläre USI Header vorangestellt und der Stream mit Prüfsumme und ETX abgeschlossen.

Name	FSP234_SW_MDS
Adresse	0xEA_H/234_D/0x4541_{ASCII}
Tiefe	dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Der interne FSP234 ist ein CPU Software FSP und liefert die MDS der an den USI gefundenen Module im ACU System.

Dem voran wird die MDS der MFU selbst gesendet.

Wird FSP234 geschrieben, startet dies einen USI Rescan. In diesem Fall werden geschickte Dummy-Daten und Prüfsumme nicht ausgewertet.

STX PID PID GW MA FSP FSP Data Data PP PP ETX

0x02 0x57 0x52 0x30 0x30 0x45 0x41 0x30 0x30 0x30 0x30 0x03

Nach einer Leseanforderung werden die im ACU System gefundenen Moduldeskriptoren wie folgt ausgegeben:

				Datenbytes für Prüfsumme											
Direkte Antwort des FSP234				MDS der MFU					MDS gefundener Module					Prüfsumme	
STX	GW	MA	FSP	USINr. (GW)	MA	Mod. FSP	MDS ACU	PP	USINr.	MA	Mod. FSP	MDS Modul	PP	PP	ETX
0x02	0x30	0x30	0x4541	0x30	0x30	0x3030	(...EOD)		0x3u	0x3m	0x3030	(...EOD)		0x.. 0x..	0x03
Einleitung mit STX, der Gateway- und Modul-, sowie die FSP Nummer				Die MDS der MFU wird wie die nachfolgenden Modul MDS mit der USI Nummer (in diesem Falle die Gatewaynummer 0, da es sich ja um die MFU handelt), der Modul- und FSP Nummer eingeleitet. Der Deskriptor wird mit dem Enddeskriptor (...EOD) und der anschließenden Prüfsumme (PP = 2 Byte ASCII) abgeschlossen. Es folgt KEIN ETX!					Die MDS der an der MFU angeschlossenen und identifizierten Module. 'u' bei USINr. liegt dabei zwischen 1...10, 'm' bei Mod.Nr. zwischen 1...8. Die einzelnen Deskriptoren werden mit dem Enddeskriptor (...EOD) und der anschließenden Prüfsumme (PP = 2 Byte ASCII) abgeschlossen, es folgt erst nach der letzten MDS ein ETX!					2 Byte ASCII Prüfsumme über alle Datenbytes	ETX nach der letzten gesendeten MDS.

Name	FSP235_SW_Logbook
Adresse	0xEB_H/235_D/0x4542_{ASCII}
Tiefe	dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Der interne FSP235 ist ein CPU Software FSP und liefert das MFU interne Logbuch.

Jeder Eintrag des Logbuchs ist 8 Byte lang.

Die ersten 5 Bytes beinhalten das Datum und die Uhrzeit des Eintrags.

Die RTC liefert die Daten in BCD.

Im Logbuch sind diese aber aus Platzgründen in komprimiert in HEX abgelegt

Bezeichnung	Bereich	Bitdarstellung	Kurzformat
Tag	1..31	5 Bit 00001..11111	D
Monat	1..12	4 Bit 0001..1100	M
Jahr	0..99	7 Bit 0000000..1100011	Y
Stunde	0..23	5 Bit 00000..10111	H
Minute	0..59	6 Bit 000000..111011	m
Sekunde	0..59	6 Bit 000000..111011	S
Wochentag	0..6	3 Bit 000.110 (0 = Sonntag)	W

Die obige BCD Darstellung wird zur Ausgabe in die nachfolgende HEX Darstellung transferiert.

Byte	Inhalt
0	DDDD DMMM
1	MYYY YYYY
2	HHHH Hmmm
3	mmmS SSSS
4	SWWW xxxx

Die drei nachfolgenden Bytes enthalten den eigentlichen Logbucheintrag.

Byte	Inhalt	Details
5	Spezifikation	0 = normaler Eintrag
		1 = Fehler
		2 = Warnung
		3 = Interlock
6	Eintrag MSB	Nibbel 3..2
7	Eintrag LSB	Nibbel 1..0

Normal:

Nibble	Bedeutung	Bereich
3	Logbucheintrag	(0..255) _D
2		
1		
0		

Fehler:

Nibble	Bedeutung	Bereich
3	USI Nummer	0x0..0xB (0..11) _D
2	Modul Nummer	0x0..0x7 (0..7) _D
1	Fehlernummer	0x00...0xFF (1.127)
0		

Warnung:

Nibble	Bedeutung	Bereich
3	Bitposition der Warnung in Dezimal	(0..31) _D
2		
1		
0		

Interlockeintrag:

Nibble	Bedeutung	Bereich
3	USI Nummer	0x0..0xB (0..11) _D
2	Modul Nummer	0x0..0x7 (0..7) _D
1	Interlockbit	0x00...0xFF (0.255)
0		

FSP235 kann nur mit einem legitimierten USB Stick geschrieben werden. Andernfalls bekommen Schreibenforderungen eine NACK Antwort mit Fehlercode.

Ist ein legitimierter USB Stick eingesteckt und wird FSP235 geschrieben, wird das Logbuch gelöscht. In diesem Fall werden geschickte Dummy-Daten und Prüfsumme nicht ausgewertet.

STX PID PID GW MA FPS FSP Data Data PP PP ETX

0x02 0x57 0x52 0x30 0x30 0x45 0x42 0x30 0x30 0x30 0x30 0x03

Name	FSP236_SW_Delete_Errors
Adresse	0xEC_H/236_D/0x4543_{ASCII}
Tiefe	dynamisch
I/O	schreiben
Reset	0x(siehe Beschreibung) _H

Der interne FSP236 ist ein CPU Software FSP und löscht den MFU internen Fehlerspeicher.

In diesem Fall werden geschickte Dummy-Daten und Prüfsumme nicht ausgewertet.

STX PID PID GW MA FPS FSP Data Data PP PP ETX

0x02 0x57 0x52 0x30 0x30 0x45 0x43 0x30 0x30 0x30 0x30 0x03

Name	FSP237_SW_HighSpeedStream
Adresse	0xED_H/237_D/0x4544_{ASCII}
Tiefe	dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Gibt eine bestimmte, frei wählbare Anzahl von USI HighSpeed Daten mit einer ebenfalls frei wählbaren Zeitverzögerung zurück.

Dazu muss der FSP zunächst beschrieben werden mit:

STX PID PID GW MA FSP FSP

[USINr.] [Quantity] [Quantity] [Quantity] [Quantity] [Pause/Skip] PP PP ETX

Mit:

PID

WR

[USINr.]

USI Nummer (0x1..0xB)

[Quantity]

Anzahl von Werten ,n' (1..65536)

[Pause/Skip]

Wartezeit zwischen zwei Werten (0..15) in uSekunden, bzw. bei neuer MFU FW die Anzahl von Werten die zwischen zwei erfassten Werte ignoriert werden sollen. Wird der FSP anschließend gelesen, werden ,n' HighSpeed Datensätze gesendet:

STX GW MA FSP FSP

[Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0] ← Datensatz n

[Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0] ← Datensatz n-1

[..]

[Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0] ← Datensatz n-n

PP PP ETX

Ab MFU SE FW 7.4.0, bzw. MFU LE FW 7.3.0 ist die Datenerfassung synchronisiert, d.h. die erfassten Daten werden nicht mehr nach einer voreinstellbaren Zeit gepollt, sondern es wird jeder neue Messwert erfasst. [Pause] gibt jetzt nicht mehr die Zeitdauer zwischen zwei erfassten Messwerten in uSekunden an, sondern wird zu [Skip] und bestimmt die Anzahl von Messwerten die zwischen zwei erfassten Messwerten verworfen werden sollen.

Name	FSP238_SW_SnapshotHighSpeed
Adresse	0xEE_H/238_D/0x4545_{ASCII}
Tiefe	dynamisch
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Liefert einen Schnappschuss der Highspeed Daten aller aktiven und im HighSpeed Modus befindlichen USIs (1..11) zurück.

Eine komplette Antwort lautet:

```
STX GW MA FSP FSP
[USI01Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI02Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI03Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI04Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI05Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI06Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI07Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI08Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI09Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI10Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
[USI11Nr. Byte7 Byte6 Byte5 Byte4 Byte3 Byte2 Byte1 Byte0]
PP PP ETX
```

mit:

[] sofern diese USI aktiv ist. Andernfalls wird die entsprechende Zeile bei der Ausgabe übersprungen.

Name	FSP239_SW_Debug
Adresse	0xEF_H/239_D/0x4546_{ASCII}
Tiefe	65536 Byte / 524288 Bit
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Dieser FSP liefert Inhalte von Flashsektoren zurück -> für Debugzwecke

Wird der FSP beschrieben, muss der zu lesende Sektor mitgeteilt werden.

STX GW MA FSP FSP SectorNr.High SectorNr.Low PP PP ETX

Beispiel: Sektor 5 lesen

□	W	R	0	0	E	F	0	5	0	5	□
02	57	52	30	30	45	46	30	35	30	35	03

Wird der FSP gelesen liefert er die Daten des zuvor gewählten Sektors.

Nach dem Bootvorgang der MFU ist immer der Sektor 5 (Parameter) gewählt.

Name	FSP240_SW_RealTimeClock
Adresse	0xF0_H/240_D/0x4630_{ASCII}
Tiefe	7 Byte / 56 Bit
I/O	Lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Der interne FSP240 ist ein CPU Software FSP und dient dem Stellen und Auslesen der MFU internen RTC (Real Time Clock).

FSP240 kann geschrieben und gelesen werden.

Zum Beschreiben wird eine reguläre USI Schreibanforderung mit folgender Syntax an dieses FSP gesendet,

STX PID PID GW MA FSP_{hi} FSP_{lo} wDay wDay Day Day Month Month Year Year
Hour Hour Minute Minute Second Second PP PP ETX

OLD_RTC (RTC im FPGA)

[55..48]	wDay, Wochentag	(0x00..0x06, wobei Sonntag = 0)
[47..40]	Day, Tag	(0x01..0x1F → 1..31))
[39..32]	Month, Monat	(0x01..0x0C → 1..12)
[31..24]	Year, Jahr	(0x00..0x63 → 0..99)
[23..16]	Hour, Stunden	(0x00..0x17 → 0..23)
[15..8]	Minute, Minuten	(0x00..0x3B → 0..59)
[7..0]	Second, Sekunden	(0x00..0x3B → 0..59)

Jeder Einstellwert (Tag, Monat usw.) besteht aus 2 Byte ASCII mit hexadezimaler Information. Nicht benötigte Bytes wie das MSB beim Wochentag und Monat werden als 0x30 gesendet.

Wird der FSP gelesen wird die Zeitinformation in der gleichen Reihenfolge ausgegeben.

Die Millisekunden werden weder von extern geschrieben noch ausgegeben. Nach dem senden der Uhrzeitdaten vom PC werden die Millisekunden auf 0 gesetzt und die RTC gestartet.

NEW_RTC (RTC im eigenen Chip, FPGA extern)

Alle Daten sind BCD codiert.

[55..48]	wDay, Wochentag	(0x30, 0x30 .. 0x30, 0x36, → 0 .. 6, mit Sonntag = 0)
[47..40]	Day, Tag	(0x30, 0x31 .. 0x33, 0x31 → 1 .. 31))
[39..32]	Month, Monat	(0x30, 0x31 .. 0x31, 0x32 → 1 .. 12)
[31..24]	Year, Jahr	(0x30, 0x30 .. 0x39, 0x39 → 0 .. 99)
[23..16]	Hour, Stunden	(0x30, 0x30 .. 0x32, 0x33 → 0 .. 23)
[15..8]	Minute, Minuten	(0x30, 0x30 .. 0x35, 0x39 → 0 .. 59)
[7..0]	Second, Sekunden	(0x30, 0x30 .. 0x35, 0x39 → 0 .. 59)

Jeder Einstellwert (Tag, Monat usw.) besteht aus 2 Byte ASCII mit BCD Information. Das nicht benötigte MSB Byte beim Wochentag wird immer als 0x30 gesendet.

Wird der FSP gelesen wird die Zeitinformation in der gleichen Reihenfolge ausgegeben.

Erst nach dem erfolgreichen senden aller Uhrzeitdaten vom PC wird die RTC gesetzt und ggf. gestartet.

Im folgenden Beispiel wird die RTC auf 16:15:25 20.06.2012 gestellt.

02	57	52	30	30	46	30	30	33	32	30	30	36	31	32	31	36	31	35	32	30	30	35	03
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Name	FSP241_SW_BitManipulation
Adresse	0xF1_H/241_D/0x4631_{ASCII}
Tiefe	3 Byte / 24 Bit
I/O	schreiben
Reset	0x(siehe Beschreibung) _H

Der interne FSP241 ist ein CPU Software FSP und dient der Bitmanipulation der FSP in der MFU.

FSP241 kann nur geschrieben werden. Leseanforderungen bekommen eine NACK Antwort mit Fehlercode.

Zum Beschreiben wird eine reguläre USI Schreibanforderung mit folgender Syntax an dieses FSP gesendet,

```
STX PID PID GW MA FSPhi FSPlo [FSPmhi FSPmlo] [BIThi BITlo] [BITVal  
BITVal] PP PP ETX
```

Einschränkung: es können nur die Bits 0..255 manipuliert werden.

[23..16] FSP_{mhi}/FSP_{mlo}, zu manipulierendes FSP (nur beschreibbare FSP)

[15..8] BIT_{hi}/BIT_{lo}, Bitnummer (gibt die Bitposition an, beginnt bei 0...255)

[7..0] BITVal, Wertigkeit des Bit,
gleich = 0x00 → Bit löschen,
ungleich ! = 0x00 → Bit setzen

Name	FSP242_SW_CPU_Status
Adresse	0xF2_H/242_D/0x4632_{ASCII}
Tiefe	4 Byte / 32 Bit
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Der interne FSP242 ist ein CPU Software FSP und repräsentiert 'CPU_Status' Informationen.

Es können allerdings nur die Bit[7] 'CPU_STATUS_RECEIVE_SYSPARAMETERS' und Bit[31] 'CPU_STATUS_DISABLE_CIRCULAR_INTERLOCK_CHECK' seitens des PC geschrieben werden. Alle anderen Bit sind **Read Only**.

Bit	Bezeichnung	Beschreibung
[31]	CPU_STATUS_DISABLE_CIRCULAR_INTERLOCK_CHECK	Wenn '1' wird KEINE zyklische Interlockabfrage durchgeführt
[30]	CPU_STATUS_CMD_TRIGGER_SOMETHING	Wenn '1' wird das Kommando 'cCMDTriggerSomething' von 'inst_MFUSwitchingOperations' ausgegeben.
[29..21]		n.u.
[20]	CPU_STATUS_SYSTEM_HAS_INTERLOCKS	Wenn '1' steht mindestens ein Interlock an
[19]	CPU_STATUS_RECEIVING_SYSPARAMETERS_RAM	Wenn '1' werden gerade Parameter vom PC empfangen
[18]	CPU_STATUS_MODULES_VERIFIED	Wird '1' sofern der FSP243 erfolgreich die gefundenen mit den gewünschten Modulen verglichen hat.
[17]	CPU_STATUS_PARAMETERS_VALID	Wenn '1' sind die in der MFU geladenen Parameter gültig
[16]	CPU_STATUS_LOADING_INTERNAL_PARAMETERS	Wenn '1' werden gerade die MFU internen Parameter geladen
[15]	CPU_STATUS_WATCHDOG	perm. Wechsel zwischen '1' und '0' als Herzschlag der CPU Funktionalität
[14]	CPU_STATUS_BOOTSEQUENZ_COMPLETED	'1' wenn die MFU die Bootsequenz nach dem PowerUp abgeschlossen hat
[13]	CPU_STATUS_VNC1L_NOT_PROGRAMMED	'1' wenn der VNC1L (USB Controller) nicht programmiert ist
[12]	CPU_STATUS_USB_DEVICE_PERMITTED	nur wenn '1' dürfen USB-, bzw. bestimmte Menüzugriffe erfolgen, wird '1' wenn ein "zugelassener" USB Stick an die MFU angeschlossen wird
[11]	CPU_STATUS_USB_DEVICE_DETECTED	Wenn '1' wurde ein eingesteckter USB Stick an der MFU Front entdeckt
[10]	CPU_STATUS_USING_INTERNAL_PARAMETERS	Wenn '1' werden die internen Systemparameter aus dem seriellen Flash benutzt. Wenn '0' wurden diese Parameter per PC

		flüchtig überschrieben.
[9]	CPU_STATUS_ERROR_OCCURED	Wenn '1' ist ein Fehler Software aufgetreten. Dieser wird sowohl im Logbuch als auch flüchtig im Speicher vermerkt.
[8]	CPU_STATUS_WARNING_OCCURED	Wenn '1' ist eine Systemwarnung aufgetreten welche nicht zwangsläufig das Gerät abgeschaltet hat, den Benutzer jedoch darauf hinweist.
[7]	CPU_STATUS_RECORDING_SYSPARAMETERS	Beim Übergang von '0' nach '1' wird der Parametersektor im seriellen Flash der MFU gelöscht. Anschließend werden alle über den USB empfangenen USI Kommandos an im seriellen Flash gespeichert. Beim Übergang von '1' nach '0' wird der Speichervorgang abgeschlossen.
[6]	CPU_STATUS_FETCHING_INTERLOCKS	Wenn '1' globale Info darüber, dass gerade Interlocks von der CPU getestet werden
[5]	MPU_STATUS_MFU_CAN_NOT_TRANSFER_ANY_DATA_RIGHT_NOW	Wenn ,1' können gerade KEINE Daten an die MFU gesendet, bzw. von dieser gelesen werden, weil diese z.B. einen USI (Re)scan oder USB-Speicherzugriffe durchführt.
[4]	CPU_STATUS_RESET_BUTTON_ACTIVE	Wenn '1' ist die RESET Taste an der MFU Vorderseite gedrückt. Im Standardbildschirm kann nun anstelle 1/50 in 1/200 Schritten der Handsollwert geändert werden.
[3]	CPU_STATUS_STDSCREEN_ACTIVE	Der Standardbildschirm ist aktiv und wird angezeigt.
[2]	CPU_STATUS_PSU_IS_REMOTE	Das ACU System ist im Remote Betrieb.
[1]	-	n.u.
[0]	CPU_STATUS_PSU_IS_ON	Das ACU System ist eingeschaltet (Regler ist freigegeben)

Name	FSP243_SW_VerifyHWConfig_ModuleClasses
Adresse	0xF3/243_D/0x4633_{ASCII}
Tiefe	dynamisch
I/O	schreiben
Reset	0x(siehe Beschreibung) _H

Über diesen FSP lässt sich die korrekte Hardwarekonfiguration des ACU-Systems verifizieren.

Dieses FSP beinhaltet dabei die Modul(-Sub-)klassen der an den USIs erwarteten Module. Der NiosII der MFU liest dieses FSP und vergleicht dessen Inhalt mit den Modul(-Sub-)klassen der gefundenen Module. Besteht Konsistenz wird die Reglerfreigabe ermöglicht.

Die Datentiefe dieses FSP ist abhängig von der ACU Konfiguration und der Anzahl der an der MFU ange-bundenen Module.

Die in diesem FSP zu verarbeitenden Daten sind dabei wie folgt organisiert:

```
STX PID PID GW MA FSP FSP { [Head Head] [ModuleClass
ModulClass] [ModuleSubClass ModulSubClass] } PP PP ETX
```

mit

[Head] oberes Nibble USI Nummer, unteres Nibbel Modul Nummer
Bsp.: 0x12_H/0x31,0x32_{ASCII} für USI 1, Modul 2

[ModuleClass] 2 Nibble Hex (2 Byte ASCII) Modulkasse des in [Head] beschriebenen Moduls

[ModuleSubClass] 2 Nibble Hex (2 Byte ASCII) Modul-Sub-klasse des in [Head] beschriebenen Mo-
duls

{ } stetige Wiederholung obiger Beschreibung für alle Module

Wird dieses FSP gefüllt, wird sofort die gefundene Modulkonfiguration mit der in diesem FSP ge-wünschten verglichen. Ist dieser Vergleich erfolgreich, wird ‚CPU_STATUS_MODULES_VERIFIED‘ im FSP242 gesetzt.

Wird die MFU ohne Module betrieben lässt sich das Flag ‚CPU_STATUS_MODULES_VERIFIED‘ setzen, indem ein Schreibkommando an FSP243 gesendet wird, bei dem alle Parameter (Head, ModulClass, ModulSubClass) auf null (0x30) stehen.

STX	PID	GW	MA	FSP	Head	ModuleClass	ModuleSubClass	PP	ETX
0x02	0x57 0x52	0x30	0x30	0x46 0x33	0x30 0x30	0x30 0x30	0x30 0x30	0x30 0x30	0x03

Name	FSP244_SW_ChangeUSIBitrate_ChangeUSIMode
Adresse	0xF4/244_D/0x4634_{ASCII}
Tiefe	2 Byte / 16 Bit
I/O	schreiben
Reset	0x(siehe Beschreibung) _H

Über diesen FSP kann seitens des PC die Bitrate sowie der USI Modus einzelner USI konfiguriert werden.

Die Anforderung ist dabei identisch mit einer Anforderung an das Standard FSP012 zur Konfiguration der USI. Schreibt der PC direkt an einen Modul FSP012, erfolgt keine Umschaltung der USI innerhalb der MFU, da die MFU lediglich als Gateway funktioniert und die Daten zwischen PC und Modulen nicht analysiert.

Daher erfolgt die Bitratenumschaltung der USI über diesen Software FSP.

Mit diesem FSP wird die Bitraten/USI Mode Umschaltung in der MFU vorgenommen und von dieser mit dem/mit den angeschlossenen Modul(en) verhandelt, damit auch diese die gewünschte Bitrate/den gewünschten USI Mode entsprechend der Anforderung einstellen.

WICHTIG: Ist eine USI in HighSpeed, kann deren Bitrate **NICHT** direkt in HighSpeed geändert werden. Es muss zuvor HighSpeed an dieser USI deaktiviert werden. Bei der HighSpeed-Deaktivierungs-Anforderung kann aber eine Wunschbitrate für die USI-Standardkommunikation mitgeteilt werden. (Bsp.: mit Data Data = 0x30 0x38 wird HighSpeed abgeschaltet und das USI wechselt sofort auf 10MBit). Im Anschluss wird ggf. in einer weiteren Anforderung an diesen FSP die neue Bitrate/der neue USI Mode eingestellt.

Sollen an einem USI mit mehreren Modulen alle Module zeitgleich auf eine neue Bitrate geschaltet werden, wird für den Parameter Mod = 0 (0x30) gewählt.

Sollen alle Module an allen USIs zeitgleich auf eine neue Bitrate geschaltet werden, wird für den Parameter USI = 0 (0x30) gewählt.

STX PID PID GW MA FSP FSP USI Mod Data Data PP PP ETX

STX	- StartOfText	(0x02)
PID PID	- Schreib Anforderung	(0x5752)
GW	- Gateway Adresse	(0x30 – da Gateway die MFU ist)
MA	- Modul Adresse	(0x30 – da Modul die MFU ist)
FSP FSP	- FSP Nummer	(0x46, 0x34)
USI	- USI Nummer an der Bitrate/Modus geändert werden soll	(0x30, 0x31...0x42)
Mod	- Modulnummer an dem Bitrate/Modus geändert werden soll	(0x30, 0x31...0x38)
Data Data	- Bitrate/Modus (siehe Standard FSP012)	
PP PP	- Prüfsumme	
ETX	- EndOfText	(0x03)

Name	FSP245_SW_intScopeDataStream
Adresse	0xF5/245_D/0x4635_{ASCII}
Tiefe	12008 Byte / 96064 Bit (bis MFU FW 7.4.x) 6008 Byte / 48064 Bit (ab MFU FW 7.5.0)
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Dieser FSP sendet Scope Kanaldaten des internen Scopes mit

8 Kanälen (bis MFU FW 7.4.x) mit je 500 Werten und 12 Bit Tiefe

→ 6000 Byte_H, bzw. 12000 Bytes_A + einbettende STX ... PP PP ETX usw.

4 Kanälen (ab MFU FW 7.5.0) mit 500 Werten und 12 Bit Tiefe

→ 3000 Byte_H, bzw. 6000 Bytes_A + einbettende STX ... PP PP ETX usw.

Bei der Ausgabe dieser Daten wird der Triggerpunkt und -offset berücksichtigt.

D.h. die Daten werden nicht beginnend vom RAM Speicher 0, sondern beginnend vom Triggeroffset ausgegeben. An der höchsten RAM-Stelle (500) wird mit der RAM Stelle 0 fortgefahren und die Ausgabe endet bei Triggeroffset – 1.

Sollen die Daten beginnend vom RAM Speicher 0 und nicht vom Triggeroffset ausgegeben werden bitte den FSP231_SW_intScopeDataStreamReadOut benutzen.

FSP245 kann nur gelesen werden. Schreibenanforderungen bekommen eine NACK Antwort mit Fehlercode.

Bis MFU FW 7.4.x gilt folgendes:

Ausgabestring nach der Leseanforderung:

STX GW MA -FSP- -CH8 (12Bit) CH7 (12Bit) CH7 (12Bit) CH5 (12Bit)
CH4 (12Bit) CH3 (12Bit) CH2 (12Bit) CH1 (12Bit) - -PP- ETX

02 30 30 45 37 dd dd dd dd dd dd dd dd dd dd dd dd
dd dd dd dd dd dd dd dd dd dd dd pp pp 03

CHx - Daten des Kanals

Insgesamt werden 500 x **8** Kanäle übertragen. D.h. der gelb markierte Bereich wird also insgesamt 500-mal übertragen. Beginnend mit dem Wert 1 für Kanal 8 bis 1, Wert 2 für Kanal 8 bis 1 usw. beginnend beim Triggerpunkt. Es bilden also immer 8 x 12Bit = 96Bit → 12 Bytes einen Datensatz. Die Prüfsumme PP folgt nach Wert 500 für Kanal 8 bis 1 über die gesamten Daten.

Ab MFU FW 7.5.0 gilt folgendes:

Ausgabestring nach der Leseanforderung:

STX GW MA -FSP- -CH4 (12Bit) CH3 (12Bit) CH2 (12Bit) CH1 (12Bit) - -PP-
ETX

02 30 30 45 37 dd dd dd dd dd dd dd dd dd dd pp pp
03

CHx - Daten des Kanals

Insgesamt werden 500 x **4** Kanäle übertragen. D.h. der gelb markierte Bereich wird also insgesamt 500-mal übertragen. Beginnend mit dem Wert 1 für Kanal 4 bis 1, Wert 2 für Kanal 4 bis 1 usw. beginnend beim Triggerpunkt. Es bilden also immer 4 x 12Bit = 48Bit → 6 Bytes einen Datensatz. Die Prüfsumme PP folgt nach Wert 500 für Kanal 4 bis 1 über die gesamten Daten.

Name	FSP246_SW_Recorded_Supplies
Adresse	0xF6/246_D/0x4636_{ASCII}
Tiefe	10800 Byte
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Dieser FSP liefert die in der MFU überwachten Betriebsspannungen der letzten 24 Stunden. Die Spannungen werden einmal je Minute gespeichert. Die Sortierung ist dabei wie folgt: 1V2, 2V5, 3V3, 5V0, p12V0, n12V0. Der Speicher der Betriebsspannungen ist flüchtig, d.h. nach dem Aus- und Wiedereinschalten sind alle Werte zunächst 0.

Je Spannung wird durch 12 Bit Daten + Vorzeichen repräsentiert. Bei der Ausgabe werden je Spannung 16 Bit gesendet. Das MSB ist dabei je Spannung das Vorzeichen, danach folgen die 12 Bit Spannung. Die restlichen 3 Bit je Spannungswert bleiben immer 0.

FSP246 kann nur gelesen werden. Schreibenanforderungen bekommen eine NACK Antwort mit Fehlercode.

Ausgabestring nach der Leseanforderung:

```
STX GW MA -FSP- 1V2 (16Bit) 2V5 (16Bit) 3V3 (16Bit) 5V0 (16Bit)
12V0 (16Bit) -12V0 (16Bit) (...) PP PP ETX
```

```
02 30 30 46 36 dd dd dd dd dd dd dd dd dd dd dd dd dd dd
dd dd dd dd dd dd dd dd dd dd 03
```

dd dd dd dd_H entspricht dabei vsss ssss ssss sxxx_B mit

v = Vorzeichen

s = Spannungswert

x = don't care, immer ,0'

Insgesamt werden 24x60=1440 Werte je Spannung übertragen. D.h. der gelb markierte Bereich wird insgesamt 1440-mal übertragen. Beginnend mit dem Wert für 1,2 Volt. Die Prüfsumme PP folgt nach dem 1440-sten Wert für -15,0 Volt über die gesamten Daten.

1440 Spannungen * 6 Einträge mit je 16 Bit 17280 Byte Hex, bzw. 34560 Byte ASCII an Daten.

Die Bitwertigkeiten sind wie folgt:

bei 1V2, 2V5, 3V3 und 5V0 werden die gemessenen Werte direkt mit dem LSB Wert (SUP_LSB_Size = 0.002441) multipliziert und ergeben so die Spannung. Bei -12V0 und 12V0 erfolgt ebenfalls zuerst die LSB Wert Multiplikation und anschließend die Multiplikation mit dem Faktor 11, der dem vorgeschalteten Spannungsteiler entspricht. Die tatsächlich gemessenen Spannungen liegen nämlich bei 1/11 der Originalspannung.

Name	FSP247_SW_Recorded_Temperatures
Adresse	0xF7/247_D/0x4637_{ASCII}
Tiefe	8640 Byte
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Dieser FSP liefert die in der MFU überwachten Modultemperaturen der letzten 24 Stunden. Die Temperaturen werden einmal je Minute gespeichert. Die Sortierung ist dabei wie folgt: FPGA, Längsregler, Modulmitte. Der Speicher der Temperaturenaufzeichnung ist flüchtig, d.h. nach dem Aus- und Wiedereinschalten sind alle Werte zunächst 0.

Je Temperatur steht ein 8 Bit, also 1 Byte Speicher zur Verfügung. Alle 3 Temperaturen belegen also pro Eintrag 3 Byte Hex, bzw. 6 Byte ASCII.

FSP247 kann nur gelesen werden. Schreibenanforderungen bekommen eine NACK Antwort mit Fehlercode.

Ausgabestring nach der Leseanforderung:

```
STX GW MA -FSP- T1(8Bit) T2(8Bit) T3(8Bit) (...) PP PP ETX
02 30 30 46 37 dd dd dd dd dd dd 03
```

Insgesamt werden 24x60=1440 Werte je Temperatur übertragen. D.h. der gelb markierte Bereich wird insgesamt 1440-mal übertragen. Beginnend mit dem Wert T1 für das FPGA. Die Prüfsumme PP folgt nach dem 1440-sten Wert für T3 über die gesamten Daten.

1440 Übertragungen mit je 3 Byte Hex sind 4320 Byte Hex, bzw. 8640 Byte ASCII an Daten.

Die Bitwertigkeiten sind wie folgt:

Aktuelle Temperatur	Gemessene Temperatur	Binär Hexadzial
+130.00°C	+127°C	0111 1111
+127.00°C	+127°C	0111 1111
+126.50°C	+126°C	0111 1110
+25.25°C	+25°C	0001 1001
+0.50°C	0°C	0000 0000
+0.25°C	0°C	0000 0000
0.00°C	0°C	0000 0000
-0.25°C	-1°C	1111 1111
-0.50°C	-1°C	1111 1111
-0.75°C	-1°C	1111 1111
-1.00°C	-1°C	1111 1111
-25.00°C	-25°C	1110 0111
-25.25°C	-26°C	1110 0110
-54.75°C	-55°C	1100 1001
-55.00°C	-55°C	1100 1001
-65.00°C	-65°C	1011 1111

Name	FSP248_SW_ReadExtRAMData
Adresse	0xF8_H/248_D/0x4638_{ASCII}
Tiefe	dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Gibt eine bestimmte, frei wählbare Anzahl von RAM Einträgen eines ExtRAMExtension Moduls zurück.

Dazu müssen zunächst Daten gesammelt worden sein.

Die Daten können nur 1x ausgelesen werden. Danach muss ein neuer Speicherzyklus durchlaufen werden. Vor dem Starten eines neuen Speicherzyklus müssen ALLE Daten ausgelesen worden sein.

Zum Auslesen der Daten muss der FSP ggf. zunächst beschrieben werden mit:

```
STX PID PID GW MA FSP FSP [Quantity][Quantity][Quantity][Quantity] PP
PP ETX
```

Mit:

```
PID PID WR
[Quantity] Anzahl von Werten (1..65536)
```

Der Resetwert von [Quantity] ist 0x0000, dadurch werden alle im ExtRAMmodul befindlichen Werte zurück geliefert.

Sofern [Quantity] != 0x0000 gesetzt wurde, lässt [Quantity] sich nach jedem ausgelesenen Datenblock beliebig ändern und auch wieder auf 0x0000 setzen um die restlichen Daten in einem Aufruf zu lesen.

Wird der FSP anschließend gelesen, werden mit jeder Leseaufforderung [Quantity] Datensätze des ExtRAMExtension Moduls gesendet. Ist [Quantity] = 0 werden alle Daten ohne Unterbrechung gesendet:

```
STX GW MA FSP FSP
[Byte3 Byte2 Byte1 Byte0] ← Datensatz n
[Byte3 Byte2 Byte1 Byte0] ← Datensatz n-1
[... ]
[Byte3 Byte2 Byte1 Byte0] ← Datensatz n-n
PP PP ETX
```

Name	FSP249_Local_Setvalue_Scaling_Factor
Adresse	0xF9_H/249_D/0x4639_{ASCII}
Tiefe	2 Byte /16 Bit
I/O	lesen / schreiben
Reset	0x0002 _H

Definiert das Teilverhältnis der Lokalen, manuellen Sollwertvorgabe.

Die Standardschrittweite bei der Sollwertvorgabe beträgt 1/50 des Nennwerts. Bei Betätigen der RESET-Taste im Local-Modus wird diese Schrittweite weiter heruntergeteilt.

:2, :4(Default), :8, :16, :32, :64, :128, ... (usw.)

Weil bei der Berechnung des Teilerfaktors dieser einfach geschoben wird, muss 'setvalue_stepwidth' die Anzahl der Bit-Schübe beinhalten.

Der Standardwert ist hier 2, dadurch wird 1/50 zu 1/200.

STX PID PID GW MA FSP FSP [Shift][Shift] PP PP ETX

Mit:

[Shift] Anzahl von Bitschüben

Name	FSP250_NIOS_SW_Version
Adresse	0xFA_H/250_D/0x4641_{ASCII}
Tiefe	dynamisch
I/O	lesen
Reset	0x(siehe Beschreibung) _H

Dieser SW FSP beinhaltet die NiosII SW Version für Remote Updates.

Er kann nur gelesen werden.

Nachfolgendes Beispiel zeigt den Nios Software-Stand 007.00004 an.

0	0	F	A	0	0	7	.	0	0	0	0	4	2	D		
02	30	30	46	41	30	30	37	2E	30	30	30	30	34	32	44	03

Name	FSP251_compressed_PCA_configuration_file
Adresse	0xFB_H/252_D/0x4642_{ASCII}
Tiefe	dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Dieser SW FSP beinhaltet komprimierte PCA Konfigurationsdatei.

Um eine Datei zu speichern muss zunächst ein Schreibkommando auf den FSP erfolgen, in dem die Größe der anschließend zu übertragenden *.xpc7-Konfigurationsdatei mitgeteilt wird.

```
STX PID PID GW MA FSP FSP -[Length (8 Byte)]- PP PP ETX
```

```
STX W R 0 0 F B Dateigroesse in Byte PP PP ETX
```

mit

[Length] ist 8 Byte ASCII lang, sollen z.B. 8.977_D = 2311_H Byte übertragen werden ist
 Length = 0x30 0x30 0x30 0x30 0x32 0x33 0x31 0x31

Im Falle von ACK schaltet die MFU automatisch auf einen Empfangsmodus um und erwartet nachfolgend die zuvor angekündigte Anzahl von Bytes. D.h. es werden nun solange Daten der komprimierten *.xpc7 Datei an die MFU gesendet, bis alle Daten im seriellen Flash gespeichert sind.

Die Daten werden dabei NICHT als ASCII-Zeichen, sondern wie sie vorliegen, als reine Rohdaten gesendet.

Der Empfang der Daten wird abschließend mit einem weiteren ACK, bzw. NACK quittiert.

Soll die Datei aus dem seriellen Flash gelesen werden, muss lediglich das FSP gelesen werden.

Die dabei empfangenen Daten werden in einen USI-Header, eine Prüfsumme und ein ETX eingebettet.

Die Daten an sich sind aber wie beim Senden auch, die reinen Rohdaten und NICHT ASCII kodiert.

Name	FSP252_UpdateMFU_CFI_SoftwareViaRemote
Adresse	0xFC_H/252_D/0x4643_{ASCII}
Tiefe	Dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Das CFI beinhaltet 2 Software-Versionen, die 'Factory-SW' und die 'Application-SW' und zusätzlich den Boot Copier

Via FSP252 können beide Softwares und der Boot Copier ersetzt werden.

Per Default wird aber i.d.R. nur die Application-SW ersetzt.

Dazu muss das FSP252 zunächst ein Init-Kommando erhalten:

```
STX PID PID GW MA FSP FSP -Data- PP PP ETX
```

(Update Boot Copier)

```
STX W R 0 0 F C UBC+Dateigroesse in Byte PP PP ETX
```

(Update Application SW)

```
STX W R 0 0 F C UAS+Dateigroesse in Byte PP PP ETX
```

Bei der Factory-SW muss das Init-Kommando entsprechend so lauten:

(Update Factory SW)

```
STX W R 0 0 F C UFS+Dateigroesse in Byte PP PP ETX
```

Die Dateigroesse in Byte muss immer 8 Zeichen lang sein. Die Angabe der Dateigröße erfolgt in Hexadezimal. Führende Stellen sind 0.

Nachfolgend soll als Beispiel das Application-Image ersetzt werden. Die dazu nötige *.s19-Datei hat eine Größe von 959232_D = EA300_H Bytes.

0	W	R	0	0	F	C	U	A	S	0	0	0	E	A	3	0	0	4	0	0
02	57	52	30	30	46	43	55	41	53	30	30	30	45	41	33	30	30	34	30	03

Die Prüfsumme ist hierbei über den gesamten -Data- Bereich (UAS000EA300) zu bilden.

Darauf antwortet FSP252 mit einem ACK, bzw. im Fehlerfall einem NACK.

Im Falle von ACK schaltet die MFU automatisch auf einen Empfangsmodus um und erwartet nachfolgend die zuvor angekündigte Anzahl von Bytes. D.h. es werden nun solange Daten der *.s19-Datei an die MFU gesendet, bis alle Daten im CFI gespeichert sind. Die Daten werden dabei NICHT als ASCII-Zeichen, sondern wie sie vorliegen, als reine Rohdaten gesendet.

Der Empf. der *.s19-Datei wird mit ACK, bzw. NACK quittiert.

Sollen Daten aus dem CFI gelesen werden, muss dies zuvor per Schreibzugriff definiert welche Daten genau wie viele Bytes davon gelesen werden sollen.

(Read Boot Copier)

```
STX W R 0 0 F C RBC+Dateigroesse in Byte PP PP ETX
```

(Read Application SW)

```
STX W R 0 0 F C RAS+Dateigroesse in Byte PP PP ETX
```

(Read Factory SW)

```
STX W R 0 0 F C RFS+Dateigroesse in Byte PP PP ETX
```

Anschließend erfolgt eine Leseaufforderung. Diese liefert dann die gewünschte Anzahl an Bytes.

Bsp.: 390000_D = 5F370_H Byte der Application SW lesen:

0	W	R	0	0	F	C	R	A	S	0	0	0	5	F	3	7	0	3	7	0
02	57	52	30	30	46	43	52	41	53	30	30	30	35	46	33	37	30	33	37	03

Die Prüfsumme ist hierbei über den gesamten –Data– Bereich (RAS0005F370) zu bilden.

Anschließend erfolgt ein Lesekommando. Die dabei empfangenen Daten werden in einen USI-Header, eine Prüfsumme und ein ETX eingebettet.

Die Daten an sich sind aber wie beim Senden auch, die reinen Rohdaten und NICHT ASCII kodiert.

Name	FSP253_UpdateMFU_EPCS_FirmwareViaRemote
Adresse	0xFD_H/253_D/0x4644_{ASCII}
Tiefe	dynamisch
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Das EPCS beinhaltet 2 Firmware-Versionen, das 'Factory-Image' und das 'Application-Image'.

Via FSP253 können beide Images ersetzt werden.

Per default wird aber i.d.R. nur das Application-Image ersetzt.

Dazu muss das FSP253 zunächst ein Init-Kommando erhalten:

```
STX PID PID GW MA FSP FSP -Data- PP PP ETX
```

(Update Application Image)

```
STX W R 0 0 F D UAI+Dateigroesse in Byte PP PP ETX
```

Beim Factory Image muss das Init-Kommando entsprechend so lauten:

(Update Factory Image)

```
STX W R 0 0 F D UFI+Dateigroesse in Byte PP PP ETX
```

Die Dateigroesse in Byte muss immer 8 Zeichen lang sein. Die Angabe der Dateigröße erfolgt in Hexadezimal. Führende Stellen sind 0.

Nachfolgend soll als Beispiel das Application-Image ersetzt werden. Die dazu nötige *.rbf-Datei hat eine Größe von 988394_D = F14EA_H Bytes.

0	W	R	0	0	F	D	U	A	I	0	0	0	F	1	4	E	A	2	A	0
02	57	52	30	30	46	44	55	41	49	30	30	30	46	31	34	45	41	32	41	03

Die Prüfsumme ist hierbei über den gesamten -Data- Bereich (UAI000F14EA) zu bilden.

Daraufhin antwortet FSP253 mit einem ACK, bzw. im Fehlerfall einem NACK.

Im Falle von ACK schaltet die MFU automatisch auf einen Empfangsmodus um und erwartet nachfolgend die zuvor angekündigte Anzahl von Bytes. D.h. es werden nun solange Daten der *.rbf-Datei an die MFU gesendet, bis alle Daten im EPCS gespeichert sind. Die Daten werden dabei NICHT als ASCII-Zeichen, sondern wie sie vorliegen, als reine Rohdaten gesendet

Der Empfang der *.rbf Daten wird mit ACK, bzw. NACK quittiert.

Sollen Daten aus dem EPCS gelesen werden, muss zuvor per Schreibzugriff definiert werden ob das Application oder das Factory Image gelesen werden soll und wie viele Bytes davon gelesen werden sollen.

(Read Application Image)

```
STX W R 0 0 F D RAI+Dateigroesse in Byte PP PP ETX
```

(Read Factory Image)

```
STX W R 0 0 F D RFI+Dateigroesse in Byte PP PP ETX
```

Bsp.: 988394_D = F14EA_H Byte der Application FW lesen

0	W	R	0	0	F	D	R	A	I	0	0	0	F	1	4	E	A	2	D	0
02	57	52	30	30	46	44	52	41	49	30	30	30	46	31	34	45	41	32	44	03

Die Prüfsumme ist hierbei über den gesamten -Data- Bereich (RAI000F14EA) zu bilden.

Anschließend erfolgt ein Lesekommando. Die dabei empfangenen Daten werden in einen USI-Header, eine Prüfsumme und ein ETX eingebettet.

Die Daten an sich sind aber wie beim Senden auch, die reinen Rohdaten und NICHT ASCII kodiert.

Name	FSP254_Parameter_Information_String
Adresse	0xFE_H/254_D/0x4645_{ASCII}
Tiefe	Dynamisch, max. 256 Zeichen
I/O	lesen / schreiben
Reset	0x(siehe Beschreibung) _H

Parameterinformationen in ASCII Klartext.

STX PID PID GW MA FSP FSP [Text] PP PP ETX

Mit:

[Text] max. 256 Zeichen Textinformation über die in der MFU hinterlegten Parameter

Name	FSP255_SW_Flash_VNC2
Adresse	0xFF_H/255_D/0x4646_{ASCII}
Tiefe	65536 Byte / 524288 Bit
I/O	schreiben
Reset	0x(siehe Beschreibung) _H

Der interne FSP255 ist ein CPU Software FSP und wird zum programmieren des Vinculum VNC2 USB Master uController in der MFU benutzt.

FSP255 kann nur geschrieben werden. Leseanforderungen bekommen eine NACK Antwort mit Fehlercode.

Bis MFU Software 002.00002

Zum programmieren des Vinculum VNC2 wird eine reguläre USI Schreibanforderung an dieses FSP gesendet. Die nach dem Header folgenden Daten werden von der MFU bei diesem FSP ohnehin verworfen, d.h. es brauchen weder Daten noch eine Prüfsumme vor dem ETX gesendet werden.

STX PID PID GW MA FSP FSP ETX

STX	-	StartOfText	(0x02)
PID PID	-	Schreibanforderung	(0x5752)
GW	-	Gateway Adresse	(0x30 – da Gateway die MFU ist)
MA	-	Modul Adresse	(0x30 – da Modul die MFU ist)
FSP FSP	-	FSP Nummer	(0x46, 0x46)
[Daten..Prüfsumme]	-	nicht notwendig, kann entfallen	
ETX	-	EndOfText	(0x03)

Wichtig: Sofern die Programmieranforderung von der MFU verstanden wurde, wird KEIN ACK gesendet. Wird diese hingegen nicht verstanden erfolgt das senden eines NACK.

Im Erfolgsfall wechselt die Anzeige des LCD und zeigt „Waiting for flash ROM data“.

Nun wird das ROM File gesendet. Wichtig ist hierbei, dass die Übertragung in 'Hex' erfolgt.

Erst nach dem erfolgreichen senden und programmieren des ROM Files wird von der MFU ein ACK, im Fehlerfall ein weiteres NCK gesendet.

Ab MFU Software 002.00003

Zum programmieren des Vinculum VNC2 wird eine reguläre USI Schreibanforderung an dieses FSP gesendet. Nach dem Header wird der MFU die Länge der folgenden Programmierdatei mitgeteilt. Dies ist notwendig, weil die MFU auf dem TFT eine Fortschrittsanzeige generiert und der VNC2 nur vollständig geschriebene Pages akzeptiert. Ist die Programmierdatei zu Ende aber die letzte Page des VNC2 noch nicht vollständig programmiert schlägt die gesamte Programmierung fehl. Die MFU generiert aus der Information der Programmierdatenlänge und den evtl. abschließend fehlenden Daten eine vollständige letzte Page für den VNC2 automatisch. Dazu benötigt sie aber die Länge der Programmierdaten. Bis V002.00002 war die Länge per #define festgelegt, was das Programmieren des VNC2 mit neueren Softwareständen deren Länge davon abweicht nicht möglich machte.

STX PID PID GW MA FSP FSP Length Length Length Length PP PP ETX

STX	-	StartOfText	(0x02)
PID PID	-	Schreibanforderung	(0x5752)
GW	-	Gateway Adresse	(0x30 – da Gateway die MFU ist)
MA	-	Modul Adresse	(0x30 – da Modul die MFU ist)
FSP FSP	-	FSP Nummer	(0x46, 0x46)
Length	-	Länge der nachfolgenden Programmierdatei	(0x30, 0x30, 0x30, 0x33, 0x32, 0x30, 0x39, 0x30)
PP PP	-	Prüfsumme über Length	
ETX	-	EndOfText	(0x03)

Sofern die Programmieranforderung von der MFU verstanden wurde, wird EIN ACK gesendet die Anzeige des TFT zeigt „Waiting for flash ROM data“.

Wird diese hingegen nicht verstanden erfolgt das senden eines NACK.

Nun wird das ROM File gesendet. Wichtig ist hierbei, dass die Übertragung in 'Hex' erfolgt.

Nach dem erfolgreichen senden und programmieren des ROM Files wird von der MFU ein weiteres ACK, im Fehlerfall ein NACK gesendet.