EDP-CM-STR9 CPU Module User Manual

Version v4

This document contains information on the STR9 module for the RS EDP system.

Contents

1. EDP-CM-STR9 CPU Module

The STR9 module uses development tools for ARM CPUs. The recommended tool chains are as follows...

- Keil uVISION IDE with ARM RealView compiler
- HitopARM IDE based on the GNU-C compiler.

Other toolchains may be used but there are no specific examples provided for them.

The Keil uVISION is designed to work with several types of JTAG tool, the most popular being the ULINK2 JTAG debugger. The ULINK2 tool can be used with all ARM7 variants, ARM9, Cortex M0 and Cortex M3 series cores. On later Cortex M series devices the ULINK2 can switch modes from JTAG to SWD which is the new Serial Wire Debug standard. The ULINK2 is not sold with the STR9 module but can be purchased separately from one of Kiel's distribution agent. This is the best solution for development with the STR9 module as it means only one IDE is used in the development process.
Keil uVISION can be used in both simulator mode and also in hardware debug mode, using the actual hardware to single step through the code etc.

Keil uVISION can also be used to generate an output debug file (ELF/DWARF format) which can be used by other IDEs such as HiTOP for ARM, which is Hitex's own proprietary debugger. With this in mind you can use the Keil uVISION IDE to do the code writing and editing, and then HiTOP for programming and debugging. The big advantage of doing this is that HiTOP for ARM works with a JTAG debugger which is present on the STR9 module. This is basically an FTDI chip with an on board wiggler which acts as a programmer and debug interface. This means you do not need to purchase an additional hardware debug/programmer tool as it is already built into the STR 9 module.
What is more the FTDI chip has a second uncommitted RS232 channel, which can be used by the user as a virtual comm. port. This means if you set up the STR9 module correctly and connect the UART Rx and Tx traffic to route this chip, then you can effectively get a virtual UART port for free. This means through the one mini USB connection you have both a debugger and a virtual communications port. This also means if you want to view serial traffic, you do not need to purchase a Communication Module. This is described later.

2. Get The Latest Versions

Always visit the EDP support website for the latest versions of the tools and examples. This is frequently updated and contains huge amount of useful information. Hitex currently provide an RS-EDP support page and this is on...

www.hitex.co.uk/edp

RS will also provide support for the RS-EDP platform on their own web site. It is envisaged that RS site will replace the Hitex web site and become the sole repository of information on the RS-EDP platform. No URL exists (web address) at the time of writing this manual.

The key documents that are available for the RS-EDP platform are as follow...

- **Spec Sheet**
  This is a single page document detailing the features of the module. Each AM and CM has its own spec sheet.
- **User Manuals**
  A user manual is provided for each module. This is a detailed description of the module, how to configure it plus the circuit diagram and component overlay drawings for that module.
- **Mapping Aid**
  This item describes how the modules interconnect with each other via the backplane. By examining this document it is possible to see at a glance which features of the Applications modules are accessible via the CPU Module.

- **Pin Allocation Spreadsheets**
  This document shows which pins of the MCU are allocated to which function on the RS-EDP backplane. It details more thoroughly the relationship between the MCU and the Back Plane function than the mapping aid. This document is only available for the CPU Modules
- **Software downloads**
  Example software of using the CPU Module with the other RS-EDP modules.

3. Module Features

STR912 CPU Module	Part Number EDP-CM-STR9
Features	**Comment**
ARM9 CPU at 96MHz, 3V3	ARM966 Core, 128TQFP package
512k bytes on board FLASH	
96k on board SRAM	
External 1M bit SRAM, 3V SRAM	Arranged as 64kx16, <25ns, Chipselect = CS0
1x CAN transceiver with PESD2CAN protection	Local CAN transceiver
25MHz XTAL 32kHz XTAL for RTC	CPU Runs at 96MHz 32KHz for Real Time Clock
USB ESD protection on USB +/- pins	
STE100P PHY	Ethernet 10/100M bit
1.8V core regulator	
FTDI USB-JTAG and ASC1 interface	For use with HiTOP ARM IDE
Raw JTAG connector (0.05" socket)	Samtec FTSH. 0.1" JTAG adaptor available
Raw ETM trace connector (Mictor socket)	Optional fitted part for Extended Trace Macrocell
User LED, Power On LED	Yellow (Port 7.0) Blue
ResetIN LED	Orange
ResetOUT LED	Red

4. Pin Mapping

The STR9 is a high integration MCU device with 128pins. Many of the pins have more than one function for both the MCU and also within the RS-EDP system. This means that not all of the functions are available all of the time and system designer has to take care to ensure the key features of his system are implemented at the expenses of the less important ones. Some compromise has to be made by system user to accommodate the fixed mapping which you see below.
For example the virtual comm. port uses ASC1_Rx and ASC1_Tx pins which are also used by the Ethernet controller. This means for systems which use the Ethernet device, the user cannot use the virtual communications port. However serial comms is possible on ASC0 channel and therefore the user can have both Ethernet and RS232 comms with the addition of a Communication Module.

4.1 MCU Pin Allocation

STR912		Comment	RS-EDP Backplane	EDPCON	EDPCON
Pin	Name		Function	1	2
1	P4.2	3 hard wired	AN2	5	
		3 hard wired	EVG3_GPIO46	67	
		3 hard wired	EVG11_GPIO59	80	
2	P4.1	2 hard wired	AN1	4	
		2 hard wired	EVG1_GPIO42	63	
3	P4.0	4 hard wired	AN0	3	
		4 hard wired	EVG0_GPIO40	61	
		4 hard wired	EVG2_GPIO44	65	
		4 hard wired	EVG9_GPIO57	78	
4	AVSS Analog GND		VAGND	19	
5	P7.0	2 hard wired	MOTORH0_ENC0	116	
		2 hard wired	EVM2_GPIO41_CAPADC	62	
6	P7.1	2 hard wired	MOTORH1_ENC1	118	
		2 hard wired	EVM3_GPIO43	64	
7	P7.2	2 hard wired	MOTORH2_ENC2	120	
		2 hard wired	EVM4_GPIO45	66	
8	VSSQ - GND for I/O		SGND	131	
9	VDDQ 3.3 I/O		3.3V	127	
10	P2.0	2 hard wired	I2CGEN1_SCL	119	
		2 hard wired	CNTRL_I2C_SCL		79 & 80
11	P2.1	2 hard wired	I2CGEN1_SDA	117	
		2 hard wired	CNTRL_I2C_SDA		77 & 78
12	P5.0		ASC0_TX_TTL	91	
13	P7.3		EVM5_GPIO47	68	
14	P7.4		IRQ_GPIO22_I2C_INT	43	
15	P7.5		IRQ_GPIO20_I2C_GEN1_INT	41	
16	VSS - GND for CPU		SGND	131	
17	VDD - 1.8V CPU		3.3V	127	
18	P5.1		ASC0_RX_TTL	89	
19	P6.2	3 hard wired	MOTORP1H	106	
		3 hard wired	CPU_DAC01_GPIO19	40	
		3 hard wired	EVG14_GPIO62	83	
20	P6.3	2 hard wired	MOTORP1L	104	
		2 hard wired	EVG16_GPIO64	85	
21	EMI_BWR_WRLn		#WR		47 & 48
22	EMI_WRHn		#WRH		49 & 50
23	VDDQ 3.3 I/O		3.3V	127	
24	VSSQ - GND for I/O		SGND	131	

25	PHYCLK & P5.2		local Ethernet		

Not all of the IO pins on the MCU are mapped to the backplane. Some pins use the local resources present on the STR9 module.

In the tables below you will find that some of the backplane functions are hardwired to each other. Where this is the case the phrase 'hard wired' is used. This means these backplane functions are not independent of each other and cannot be separated. The user must therefore be careful on contention between Application Modules, which may effectively try and independently use these signals. Where the user can separate the functions, the phrase 'link options' is used. Where the situation is referred to as 'options' it means the situation is more complicated. You will need to refer to the Mappping Aids or the circuit diagram for more details on how the pins are used and connected.

	STR912	Comment	RS-EDP Backplane	EDPCON	EDPCON
Pin	Name		Function	1	2
26	P8.0	3 options	GPIO0	21	
		3 options	GPIO38_AD0	59	
		3 options	A0_AD0		41 & 42
27	P5.3		local Ethernet		
28	P8.1	3 options	GPIO1	22	
		3 options	GPIO36_AD1	57	
		3 options	A1_AD1		39 & 40
29	P6.0	3 hard wired	MOTORP0H	102	
		3 hard wired	CPU_DACO0_GPIO17	38	
		3 hard wired	EVG10_GPIO58	79	
30	P8.2	3 options	GPIO3	24	
		3 options	GPIO34_AD2	55	
		3 options	A2_AD2		37 & 38
31	P6.1	2 hard wired	MOTORP0L	100	
		2 hard wired	EVG12_GPIO60	81	
32	P8.3	3 options	GPIO6_MCI_DAT2	27	
		3 options	GPIO32_AD3	53	
		3 options	A3_AD3		35 & 36
33	P2.2		I2C_GEN0_SCL		7 & 8
34	P8.4	3 options	GPIO4_MCI_DAT1	25	
		3 options	GPIO30_AD4	51	
		3 options	A4_AD4		33 & 34
35	P2.3		I2C_GEN0_SDA		5 & 6
36	P8.5	2 link options	GPIO28_AD5	49	
		2 link options	A5_AD5		31 & 32
37	P2.4		CNTRL_SPI_CLK		69 & 70
38	P8.6	2 link options	GPIO26_AD6	47	
		2 link options	A6_AD6		29 & 30
39	VBATT - for RTC		3V BAT	124	
40	VSSQ - GND for I/O		SGND	131	
41	X2_RTC		local Xtal		
42	X1_RTC		local Xtal		
43	VDDQ 3.3 I/O		3.3V	127	
44	P8.7	2 link options	GPIO24_AD7	45	
		2 link options	A7_AD7		27 & 28
45	P2.5		CNTRL_SPI_MTSR		73 & 74
46	P9.0	2 link options	EVG0_GPIO40	61	
		2 link options	A8_AD8		25 & 26
47	P9.1	2 link options	GPIO37_AD9	58	
		2 link options	A9_AD9		23 & 24
48	VSS - GND for CPU		SGND	131	
49	VDD - 1.8V CPU				

50	P9.2	2 link options	GPIO35_AD10	56	
		2 link options	A10_AD10		21 & 22

	STR912	Comment	RS-EDP Backplane	EDPCON	EDPCON
Pin	Name		Function	1	2
51	P9.3	2 link options	GPIO33_AD11	54	
		2 link options	A11_AD11		19 & 20
52	P9.4	2 link options	GPIO31_ADI2	52	
		2 link options	A12_AD12		17 & 18
53	P2.6		CNTRL_SPI_MRST		71 & 72
54	P2.7 &b USBCLK		CNTRL_SPI_#CS_NSS		75 & 76
55	P3.0		ASC1_TX_TTL_ASC0_DTR	97	
56	VSSQ - GND for I/O		SGND	131	
57	VDDQ 3.3 I/O		3.3V	127	
58	P9.5	2 link options	GPIO29_AD13	50	
		2 link options	A13_AD13		15 & 16
59	P3.1		ASC1_RX_TTL_ASC0_DSR	99	
60	P3.2		CAN0_TX		63 &64
61	P3.3		CAN0_RX		61 & 62
62	P9.6	2 link options	GPIO27_AD14	48	
		2 link options	A14_AD14		13 & 14
63	P3.4	2 hard wired	SPI_SSC_CLK	98	
		2 hard wired	GPIO10_MCI_CLK	31	
64	P9.7	2 link options	GPIO25_AD15	46	
		2 link options	A15_AD15		11 & 12
65	P3.5	2 hard wired	SPI_SSC_MRST_MISO	94	
		2 hard wired	GPIO2_MCI_DAT0	23	
66	P3.6	2 hard wired	SPI_SSC_MISR_MOSI	96	
		2 hard wired	GPIO12_MCI_CMD	33	
67	P0.0		local Ethernet		
68	P3.7	2 hard wired	SPI_SSC_#CS_NSS	101	
		2 hard wired	GPIO8_MCI_DAT3	29	
69	P0.1	2 hard wired	EVG19_GPIO67	88	
		2 hard wired	local Ethernet		
70	P5.4		#CS0		53 & 54
71	P0.2		local Ethernet		
72	VSSQ - GND for I/O		SGND	131	
73	VDDQ 3.3 I/O		3.3V	127	
74	EMI_ALE		ALE		43 & 44
75	EMI_RDn		#RD		45 & 46

	STR912	Comment	RS-EDP Backplane	EDPCON	EDPCON
Pin	Name		Function	1	2
76	P0.3		local Ethernet		
77	P5.5		#CS1		55 & 56
78	P0.4	2 hard wired	EVM0_GPIO21	42	
		2 hard wired	local Ethernet		
79	P5.6	2 hard wired	IRQ_GPIO16_CNTRL_I2C_INT	37	
		2 hard wired	#CS2		57 & 58
80	P5.7	2 hard wired	IRQ_GPIO18_I2C_GEN0_INT	39	
		2 hard wired	#CS3		59 & 60
81	VDD - 1.8V CPU				
82	VSS - GND for CPU		SGND	131	
83	P6.4	2 hard wired	MOTORP2H	110	
		2 hard wired	EVG18_GPIO66	87	
84	P6.5	2 hard wired	MOTORP2L	108	
		2 hard wired	EVG17_GPIO65	86	
85	P0.5	2 hard wired	EVM1_GPIO23	44	
		2 hard wired	local Ethernet		
86	VDDQ 3.3 I/O		3.3V	127	
87	VSSQ - GND for I/O		SGND	131	
88	P0.6	2 hard wired	EVM10_GPIO68_ASCO_CTS	90	
		2 hard wired	local Ethernet		
89	RESET_Inn		#RESIN		1 & 2
90	P0.7	3 hard wired	local Ethernet		
		3 hard wired	GPIO14_MCI_PWR	35	
		3 hard wired	EVG20_GPIO69_ASCO_RTS	92	
91	TAMPER_IN		GPIO5_I2S_TX_WS	26	
92	P6.6	2 hard wired	MOTOR_TCO_FB	122	
		2 hard wired	EVM8_GPIO53	74	
93	P6.7	2 hard wired	EMG_TRAP	114	
		2 hard wired	EVM9_GPIO55	76	
94	MII_MDIO		local Ethernet		
95	USBDN		USB_DEV_D-		87 & 88
96	USBDP		USB_DEV_D+		85 & 86
97	JRTCK		local JTAG		
98	P1.0	3 options	local Ethernet		
		3 options	ASC1_TX_TTL	95	
		3 options	local Virtual Comms		
99	P1.1	3 options	local Ethernet		
		3 options	ASC1_RX_TTL	93	
		3 options	local Virtual Comms		
100	RESET_OUTn		#RESOUT		3 & 4

	STR912	Comment	RS-EDP Backplane	EDPCON	EDPCON
Pin	Name		Function	1	2
101	P1.2		local Ethernet		
102	VDDQ 3.3 I/O		3.3V	127	
103	X2_CPU		local Xtal		
104	X1_CPU		local Xtal		
105	VSSQ - GND for I/O		SGND	131	
106	P1.3		local Ethernet		
107	JRTRSTn		local JTAG		
108	JTCK		local JTAG		
109	P1.4		local Ethernet		
110	P1.5		local Ethernet		
111	JTMS		local JTAG		
112	VDD - 1.8V CPU				
113	VSS - GND for CPU		SGND	131	
114	P1.6		local Ethernet		
115	JTDI		local JTAG		
116	P1.7		local Ethernet		
117	JTDO		local JTAG		
118	P7.6		EVM6_GPIO49	70	
119	P7.7		EVM7_GPIO51	72	
120	VDDQ 3.3 I/O		3.3V	127	
121	VSSQ - GND for I/O		SGND	131	
122	AVDD - Analog 3.3V				
123	AVREF	2 link option	AN_REF	1	
		2 link option	3.3V	127	
124	P4.7	2 link option	AN7	10	
		2 link option	EVG8_GPIO56	77	
125	P4.6	3 link option	AN6	9	
		3 link option	EVG7_GPIO54	75	
		3 link option	EVG15_GPIO63	84	
126	P4.5	2 options	AN5	8	
		2 options	EVG6_GPIO52	73	
127	P4.4	3 options	AN4	7	
		3 options	EVG5_GPIO50	71	
		3 options	EVG13_GPIO61	82	
128	P4.3	2 options	AN3	6	
		2 options	EVG4_GPIO48	69	

4.2 Backplane Resources Used by the CPU Module

All of these signals detailed below are connected to the MCU in some way.

Resources Used/Available
Vcc_CM
#RESIN
#RESOUT
3.3V
3V BAT
AN_REF

VAGND
SGND
#CS0
#CS1
#CS2
#CS3
#RD
#WR
#WRH
ALE
A0_AD0
A1_AD1
A2_AD2
A3_AD3
A4_AD4
A5_AD5
A6_AD6
A7_AD7
A8_AD8
A9_AD9
A10_AD10
A11_AD11
A12_AD12
A13_AD13
A14_AD14
A15_AD15
AN0
AN1
AN2
AN3
AN4
AN5
AN6
AN7
CNTRL_I2C_SCL
CNTRL_I2C_SDA
I2C_GEN0_SCL
I2C_GEN0_SDA
I2C_GEN1_SCL
I2C_GEN1_SDA
IRQ_GPIO16_CNTRL_I2C_INT
IRQ_GPIO18_I2C_GEN0_INT
IRQ_GPIO20_I2C_GEN1_INT
IRQ_GPIO22_I2C_INT
EVG0_GPIO40
EVG0_GPIO40
EVG1_GPIO42
EVG2_GPIO44
EVG3_GPIO46
EVG4_GPIO48
EVG5_GPIO50
EVG6_GPIO52
EVG7_GPIO54
EVG8_GPIO56
EVG9_GPIO57
EVG10_GPIO58
EVG11_GPIO59
EVG12_GPIO60
EVG13_GPIO61
EVG14_GPIO62

EVG15_GPIO63
EVG16_GPIO64
EVG17_GPIO65
EVG18_GPIO66
EVG19_GPIO67
EVG20_GPIO69_ASCO_RTS
EVM0_GPIO21
EVM1_GPIO23
EVM2_GPIO41_CAPADC
EVM3_GPIO43
EVM4_GPIO45
EVM5_GPIO47
EVM6_GPIO49
EVM7_GPIO51
EVM8_GPIO53
EVM9_GPIO55
EVM10_GPIO68_ASCO_CTS
GPIO0
GPIO1
GPIO3
GPIO5_I2S_TX_WS
GPIO24_AD7
GPIO25_AD15
GPIO26_AD6
GPIO27_AD14
GPIO28_AD5
GPIO29_AD13
GPIO30_AD4
GPIO31_ADI2
GPIO32_AD3
GPIO33_AD11
GPIO34_AD2
GPIO35_AD10
GPIO36_AD1
GPIO37_AD9
GPIO38_AD0
CAN0_RX
CAN0_TX
ASC0_RX_TTL
ASC0_TX_TTL
ASC1_RX_TTL
ASC1_TX_TTL
ASC1_RX_TTL_ASC0_DSR
ASC1_TX_TTL_ASC0_DTR
USB_DEV_D-
USB_DEV_D+
MOTORP0H
MOTORP0L
MOTORP1H
MOTORP1L
MOTORP2H
MOTORP2L
MOTORH0_ENC0
MOTORH1_ENC1
MOTORH2_ENC2
EMG_TRAP
MOTOR_TCO_FB
GPIO10_MCI_CLK
GPIO12_MCI_CMD
GPIO14_MCI_PWR

GPIO2_MCI_DAT0
GPIO4_MCI_DAT1
GPIO6_MCI_DAT2
GPIO8_MCI_DAT3
SPI_SSC_#CS_NSS
SPI_SSC_CLK
SPI_SSC_MISR_MOSI
SPI_SSC_MRST_MISO
CNTRL_SPI_#CS_NSS
CNTRL_SPI_CLK
CNTRL_SPI_MRST
CNTRL_SPI_MTSR
CPU_DACO0_GPIO17
CPU_DACO1_GPIO19

4.3 Backplane Signal Names and Connections

A cross reference of the signal name to the connections on EDPCON1 and EDPCON2 connectors are shown below. For checking with a scope, then use the Break Out connectors P601, P602, andP603 as they facilitate easier probing.

Base Board Signal Name	EDPCON1	EDPCON2	Break Out Connector	
#CS0		53 & 54		
#CS1		55 & 56		
#CS2		57 & 58		
#CS3		59 & 60		
#PSEN		51 & 52		
#RD		45 & 46		
#RESIN		1 & 2	P603	26
#RESOUT		3 & 4	P603	27
#WR		47 & 48		
#WRH		49 & 50		
12V	133		P603	47
12V	134		P603	47
12V	135		P603	47
12V	136		P603	47
12V GND	137		P603	48
12V GND	138		P603	48
12V GND	139		P603	48
12V GND	140		P603	48
3.3V	127		P603	44
3.3V	128		P603	44
3.3V		95 & 96	P603	44
3V BAT	124		P603	42
5.0V	129		P603	45
5.0V	130		P603	45
5.0V		97 & 98	P603	45
A0_AD0		41 & 42		
A1_AD1		39 & 40		
A2_AD2		37 & 38		
A3_AD3		35 & 36		
A4_AD4		33 & 34		
A5_AD5		31 & 32		
A6_AD6		29 & 30		
A7_AD7		27 & 28		
A8_AD8		25 & 26		
A9_AD9		23 & 24		

A10_AD10		21 & 22		
A11_AD11		19 & 20		
A12_AD12		17 & 18		
A13_AD13		15 & 16		
A14_AD14		13 & 14		
A15_AD15		11 & 12		
ALE		43 & 44		
AN_REF	1		P601	6
AN0	3		P603	2
AN1	4		P603	6
AN2	5		P603	1
AN3	6		P603	5
AN4	7		P602	2
AN5	8		P602	4
AN6	9		P602	1
AN7	10		P602	3
AN8	11		P601	2
AN9	12		P601	4
AN10	13		P601	1
AN11	14		P601	3
AN12	15		P603	4
AN13	16		P602	6
AN14	17		P603	3
AN15	18		P602	5
ASC0_RX_TTL	89		P602	30
ASC0_TX_TTL	91		P602	31
ASC1_RX_TTL	93		P602	32
ASC1_RX_TTL_ASC0_DSR	99		P602	35
ASC1_TX_TTL	95		P602	33
ASC1_TX_TTL_ASC0_DTR	97		P602	34
CAN0_RX		61 & 62		
CAN0_TX		63 &64		
CAN1_RX	121		P602	46
CAN1_TX	123		P602	47
CANH0		89 & 90	P603	40
CANL0		91 & 92	P603	41
CNTRL_I2C_SCL		79 & 80	P603	35
CNTRL_I2C_SDA		77 & 78	P603	34
CNTRL_SPI_#CS_NSS		75 & 76	P603	33
CNTRL_SPI_CLK		69 & 70	P603	30
CNTRL_SPI_MRST		71 & 72	P603	31
CNTRL_SPI_MTSR		73 & 74	P603	32
CPU_DACO0_GPIO17	38		P603	7
CPU_DACO1_GPIO19	40		P601	7
EMG_TRAP	114		P601	44
ETH_LNK_LED	111		P602	41
ETH_RX-	109		P602	40
ETH_RX_LED	113		P602	42
ETH_RX+	107		P602	39
ETH_SPD_LED	115		P602	43
ETH_TX-	105		P602	38
ETH_TX+	103		P602	37
EVG0_GPIO40	61		P602	16
EVG1_GPIO42	63		P602	17
EVG2_GPIO44	65		P602	18
EVG3_GPIO46	67		P602	19
EVG4_GPIO48	69		P602	20
EVG5_GPIO50	71		P602	21
EVG6_GPIO52	73		P602	22
EVG7_GPIO54	75		P602	23

EVG8_GPIO56	77		P602	24
EVG9_GPIO57	78		P601	26
EVG10_GPIO58	79		P602	25
EVG11_GPIO59	80		P601	27
EVG12_GPIO60	81		P602	26
EVG13_GPIO61	82		P601	28
EVG14_GPIO62	83		P602	27
EVG15_GPIO63	84		P601	29
EVG16_GPIO64	85		P602	28
EVG17_GPIO65	86		P601	30
EVG18_GPIO66	87		P602	29
EVG19_GPIO67	88		P601	31
EVG20_GPIO69_ASCO_RTS	92		P601	33
EVM0_GPIO21	42		P601	8
EVM1_GPIO23	44		P601	9
EVM2_GPIO41_CAPADC	62		P601	18
EVM3_GPIO43	64		P601	19
EVM4_GPIO45	66		P601	20
EVM5_GPIO47	68		P601	21
EVM6_GPIO49	70		P601	22
EVM7_GPIO51	72		P601	23
EVM8_GPIO53	74		P601	24
EVM9_GPIO55	76		P601	25
EVM10_GPIO68_ASCO_CTS	90		P601	32
GPIO0	21		P603	13
GPIO1	22		P603	15
GPIO2_MCI_DAT0	23		P603	14
GPIO3	24		P603	16
GPIO4_MCI_DAT1	25		P603	17
GPIO5_I2S_TX_WS	26		P603	19
GPIO6_MCI_DAT2	27		P603	18
GPIO7_I2S_RX_CLK	28		P603	20
GPIO8_MCI_DAT3	29		P603	22
GPIO9_I2S_RX_WS	30		P603	21
GPIO10_MCI_CLK	31		P603	23
GPIO11_I2S_RX_SDA	32		P603	24
GPIO12_MCI_CMD	33			
GPIO13_I2S_TX_CLK	34		P603	25
GPIO14_MCI_PWR	35		P603	12
GPIO15_I2S_TX_SDA	36		P603	8
GPIO24_AD7	45		P602	8
GPIO25_AD15	46		P601	10
GPIO26_AD6	47		P602	9
GPIO27_AD14	48		P601	11
GPIO28_AD5	49		P602	10
GPIO29_AD13	50		P601	12
GPIO30_AD4	51		P602	11
GPIO31_ADI2	52		P601	13
GPIO32_AD3	53		P602	12
GPIO33_AD11	54		P601	14
GPIO34_AD2	55		P602	13
GPIO35_AD10	56		P601	15
GPIO36_AD1	57		P602	14
GPIO37_AD9	58		P601	16
GPIO38_AD0	59		P602	15
GPIO39_AD8	60		P601	17
I2C_GEN0_SCL		7 & 8	P603	29
I2C_GEN0_SDA		5 & 6	P603	28
I2C_GEN1_SCL	119		P602	45
I2C_GEN1_SDA	117		P602	44

IRQ_GPIO16_CNTRL_I2C_INT	37		P603	11
IRQ_GPIO18_I2C_GEN0_INT	39		P603	10
IRQ_GPIO20_I2C_GEN1_INT	41		P603	9
IRQ_GPIO22_I2C_INT	43		P602	7
MOTOR_TCO_FB	122		P601	48
MOTORH0_ENC0	116		P601	45
MOTORH1_ENC1	118		P601	46
MOTORH2_ENC2	120		P601	47
MOTORP0H	102		P601	38
MOTORP0L	100		P601	37
MOTORP1H	106		P601	40
MOTORP1L	104		P601	39
MOTORP2H	110		P601	42
MOTORP2L	108		P601	41
MOTORPWM	112		P601	43
SGND	131		P603	46
SGND	132		P603	46
SGND		9 & 10	P603	46
SGND		99 & 100	P603	46
SPI_SSC_#CS_NSS	101		P602	36
SPI_SSC_CLK	98		P601	36
SPI_SSC_MRST_MISO	94		P601	34
SPI_SSC_MTSR_MOSI	96		P601	35
USB_DEBUG_D-		67 & 68		
USB_DEBUG_D+		65 & 66		
USB_DEV_D-		87 & 88	P603	39
USB_DEV_D+		85 & 86	P603	38
USB_HOST_D-		83 & 84	P603	37
USB_HOST_D+		81 & 82	P603	36
VAGND	19		P601	5
VAGND	20		P601	5
Vcc_CM	125		P603	43
Vcc_CM	126		P603	43
Vcc_CM		93 & 94	P603	43

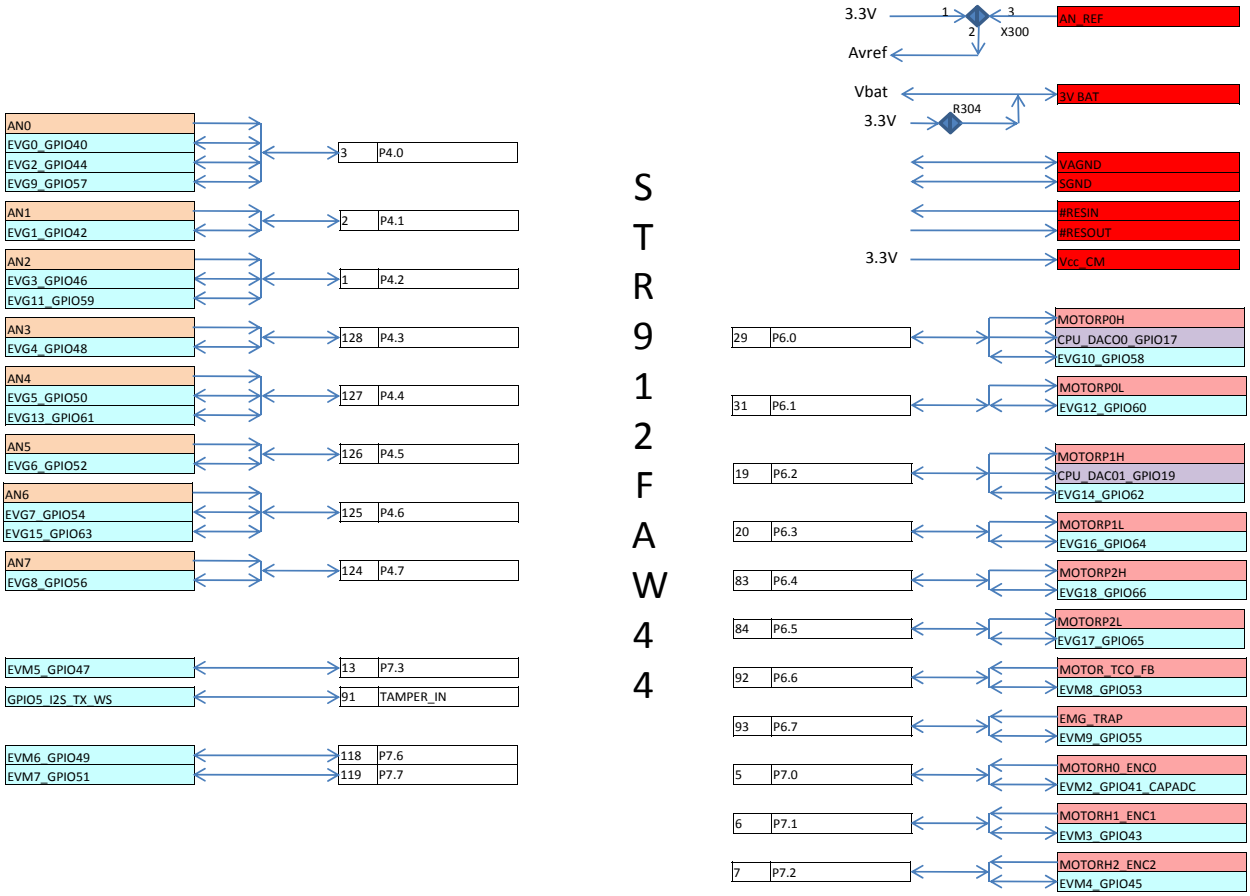
4.4 Alphabetical Listing of MCU Pins

Detailed below is the STR9 pin name cross referenced to its pin number. This facilitates the fast finding of a pin with respect to its name.

Alphabetical IO Pin Listing	
Pin	**Name**
122	AVDD - Analog 3.3V
123	AVREF
4	AVSS Analog GND
74	EMI_ALE
21	EMI_BWR_WRLn
75	EMI_RDn
22	EMI_WRHn
97	JRTCK
107	JRTRSTn
108	JTCK
115	JTDI
117	JTDO
111	JTMS
94	MII_MDIO
67	P0.0
69	P0.1

71	P0.2
76	P0.3
78	P0.4
85	P0.5
88	P0.6
90	P0.7
98	P1.0
99	P1.1
101	P1.2
106	P1.3
109	P1.4
110	P1.5
114	P1.6
116	P1.7
10	P2.0
11	P2.1
33	P2.2
35	P2.3
37	P2.4
45	P2.5
53	P2.6
54	P2.7 &b USBCLK
55	P3.0
59	P3.1
60	P3.2
61	P3.3
63	P3.4
65	P3.5
66	P3.6
68	P3.7
3	P4.0
2	P4.1
1	P4.2
128	P4.3
127	P4.4
126	P4.5
125	P4.6
124	P4.7
12	P5.0
18	P5.1
27	P5.3
70	P5.4
77	P5.5
79	P5.6
80	P5.7
29	P6.0
31	P6.1
19	P6.2
20	P6.3
83	P6.4
84	P6.5
92	P6.6
93	P6.7
5	P7.0
6	P7.1
7	P7.2
13	P7.3
14	P7.4
15	P7.5
118	P7.6

119	P7.7
26	P8.0
28	P8.1
30	P8.2
32	P8.3
34	P8.4
36	P8.5
38	P8.6
44	P8.7
46	P9.0
47	P9.1
50	P9.2
51	P9.3
52	P9.4
58	P9.5
62	P9.6
64	P9.7
25	PHYCLK & P5.2
89	RESET_Inn
100	RESET_OUTn
91	TAMPER_IN
95	USBDN
96	USBDP
39	VBATT - for RTC
17	VDD - 1.8V CPU
49	VDD - 1.8V CPU
81	VDD - 1.8V CPU
112	VDD - 1.8V CPU
9	VDDQ 3.3 I/O
23	VDDQ 3.3 I/O
43	VDDQ 3.3 I/O
57	VDDQ 3.3 I/O
73	VDDQ 3.3 I/O
86	VDDQ 3.3 I/O
102	VDDQ 3.3 I/O
120	VDDQ 3.3 I/O
16	VSS - GND for CPU
48	VSS - GND for CPU
82	VSS - GND for CPU
113	VSS - GND for CPU
8	VSSQ - GND for I/O
24	VSSQ - GND for I/O
40	VSSQ - GND for I/O
56	VSSQ - GND for I/O
72	VSSQ - GND for I/O
87	VSSQ - GND for I/O
105	VSSQ - GND for I/O
121	VSSQ - GND for I/O
104	X1_CPU
42	X1_RTC
103	X2_CPU
41	X2_RTC



Mapping Aid for IO Pins

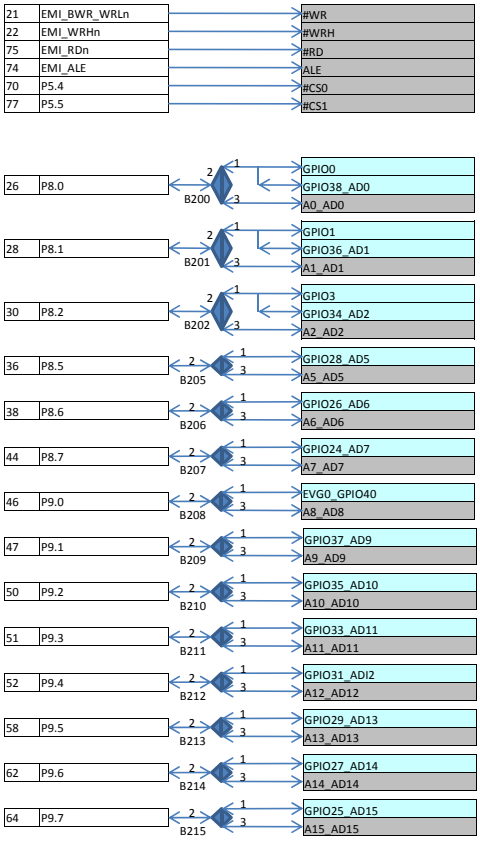
Note: On the STR9 module, the external 1M byte SRAM is mapped in to the external memory space. The options to select Ax_ADx pins (B200 – B215) is not required to use this SRAM.

The link options only provide the option to put this external bus onto the RS-EDP back plane . You would only do this if you have an external module that requires mapping into the external address space of the STR9 device.

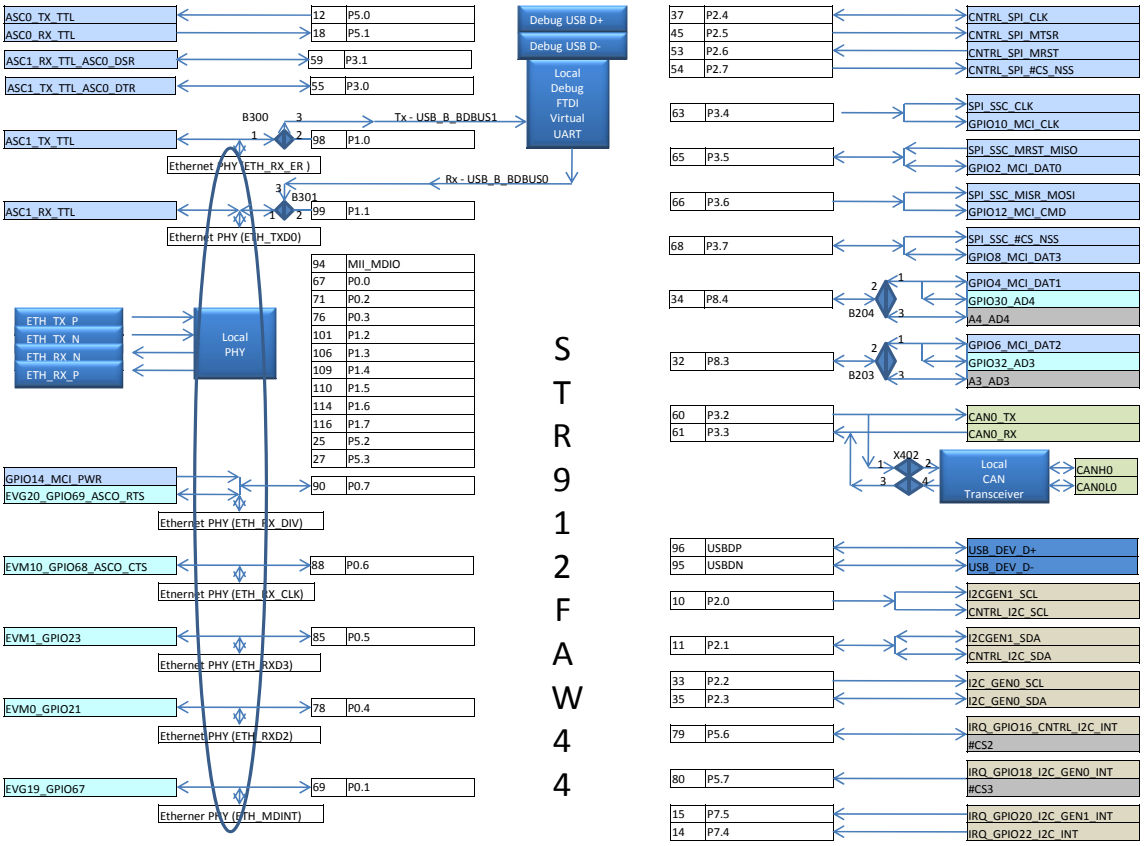
If you intend to use the on board SRAM and do NOT have an external module that requires mapping into the external bus then simply leave these jumpers open.

Note the A3_AD3 and A4_AD4 are shown on the next foil as the pins are shared with GPIO4_MCI_DAT1 and GPIO6_MCI_DAT2 which form part of the smart card interface.

STR912FAW44



Mapping Aid for IO Pins



Mapping Aid for IO Pins

4.5 Mapping Aids

The RS-EDP platform is quite a complex system as there are many modules which can plug in to it. To assist with determining how Command Modules can talk to Applications Modules a document is available called a Mapping Aid. Each Command Module has its own Mapping Aid. The Mapping aid shows diagrammatically what resources are available on the MCU and how it is connected down to the base board. The Mapping Aid also shows how the AMs are connected to the base board also. Some of the Mapping Aid details are shown above.

With this in mind the Mapping Aids can quickly establish which I/O pins on the MCU can be allocated to the various AM functions. When starting an RS-EDP design it is therefore worth spending some time allocating the I/O resources of the MCU to the various modules.

The provided software for the Command Modules has already made some assumptions as to which resources are used to connect to the AMs. The software provides all of the low level drivers needed to get the AMs working correctly.
Note: Some of the jumper and link settings for the AMs may interfere with each other as the backplane resources may effectively be used by more than one module. With this in mind the user should decide which modules he wants and check the various link options before plugging in and powering up the boards.

5. STR9 Module Selectable Jumpers

The STR9 module has the following selectable options:

Option	Type	Purpose	Default	Notes
B200-B215	Solder bridge	Connect CPU P0 multiplexed address bus to EDPCON2 or EDPCON1	Connect to EDPCON1 for IO use (ST912 bus isolated from backplane)	1-2 position – General purpose IO usage (default) 2-3 position – The external bus is made available on EDPCON2
B300-B301	Solder bridge	Connect UART1 RX & TX to USB virtual COMport	Connect UART1 to EDPCON1	1-2 position – Alows the STr9 modfule to use the standard ASC1_RX and ASC1_TX signals on the backplane or the local Ethernet PHY. 2-3 Allows the STR912 UART1 to be routed to a COMport on PC without using RS232. The virtual comm. port is available via the mini USB port on the base board.
X300	Jumper	Select source of analog reference voltage	Reference derived from local 3V3	1-2 position – local 3.3V supply (default) 2-3 position – external ANREF signal EDP-AM-AN16 module required to use external reference
X401	Jumper	Enable/disable STE100 Ethernet PHY	Closed =. Ethernet enabled	Inserted – PHY in use Removed – PHY in power down Note: UART1 lost
X402	Jumper	Enable/disable local CAN transceiver	Closed = use local transceiver	Open if opto-isolated CAN is used on EDP-AM-CO1 module. See below

5.1 B200-B125 Solder Bridges

The external bus on the STR9 MCU is made available to a local on board SRAM device. The device is a 1Mbit SRAM organised in a x16 arrangement to allow fast access to off board variables and data. By default the boards are populated with the SRAM and so the provided software is configured to address the external bus. The SRAM is always connected to the bus of the MCU irrespective of the positions of the B200-B215 solder bridges. This means the user must manage the chip select CS0 line carefully as if the user does not intend to use the SRAM at all, it must be made inert by ensuring the CS0 line remains high. The CS0 line can be used as a general purpose IO line if the external bus is not setup in software.
When the external bus is in use, the user can elect if he so wishes to make all of the necessary address, data and bus control signals, available on the EDPCON2 bus. To do this the user must ensure the solder bridges are in the 2-3 positions and the shorting links between positions 1-2 are cut. The user should only ever do this if he has designed his own module that required this external bus to be made available to him. Under normal operation the user would not do this as none of the basic EDP Application Modules require the external bus.
By leaving the shorting links in position 1-2 which is the default position the STR9 can effectively use some of the other backplane resources as general purpose IO. (assuming the external bus is not configured in software).

5.2 X300 Analog Reference Voltage

The on board A to D converter (ADC) can make use of an external stable voltage to better achieve higher accuracy results. The user has the option to select between the external ANREF signal present on the RS-EDP backplane or the local 3.3V supply of the MCU.
The AN16 Analogue module can provide either a 5.0V voltage reference or a 3.3V voltage reference (link option on the analog module) both provided from stable voltage reference devices. If the Analog Module is not fitted the user can elect to use a local 3.3V voltage source instead.

5.3 B300 & B301 Virtual Comms Ports

B300 and B301 control the ASC1_Tx and ASC1_Rx signals respectively to the virtual communication port made available through the on board FTDI chip. This chip is used as a bridge between the IDE (HiTOP) and the STR9 device and provides support for programming and debugging. The FTDI chip has an additional function also in so much as it has a virtual communication port. The user can elect to stream RS232 TTL traffic via this interface thereby allowing the user to receive standard RS232 terminal traffic via the mini USB socket on the Base Board. To enable the virtual comm. port setting the user bust select options 2-3 for both B300 and B301.

Both of these IO lines on the MCU are used by the Ethernet peripheral so it is not possible to use the Ethernet and the virtual comms port facility at the same time.

The virtual communication port will require the installation of a virtual comms port driver for the FTDI chip. This should be part of the HiTOP installation which uses the FTDI as a debugger/wiggler.

5.4 X401 Ethernet Enable/Disable

The local Ethernet PHY present on the STR9 module can be disabled with the X401 jumper. If the user does not intend to use the Ethernet capability of the STR9 device he should remove jumper X401. If he wishes to use the PHY device then the jumper should be left inserted.

5.5 X402 Local CAN Transceiver

The single CAN peripheral on the STR9 device outputs TTL level traffic, CAN0_TX and CAN0_RX This TTL traffic is routed down the backplane to the Communications Module. If the user wants an isolated CAN solution he can select via jumper options on the Communication Module to select this CAN0_TX and CAN0_RX TTL level traffic as a source. The isolated physical layer CAN output is via a pin header on the Communication Module

In addition to this the user can optionally select the traffic to routed via the on board local CAN transceiver on the STR9 CPU Module. The local CAN transceiver outputs its physical layer traffic called CANH and CANL down the back plane. This is also picked up on the Communication Module where the traffic exits the Application Module (AN) via the 9 way connector.

The user cannot use both isolated CAN and the local transceiver device at the same time as there will effectively be two CAN_RX signals which will contend with each other. The user must therefore select between isolated CAN, in which case the CAN transceiver on the Communications module is used, or the non isolated CAN in which case transceiver on the CPU Module is used. The user must therefore select the jumper options with care on both the STR9 Module and the Communications Module.

To select the non isolated CAN solution using the local CAN transceiver the user must select
X402 position 1-2 shorted and position 3-4 shorted.
Communication Module – P205 – all jumpers removed

For isolated CAN using the CAN transceiver on the Communications Module the user must select
X402 – all jumpers removed
Communications Module – P205 1-3 shorted and 4-6 shorted

5.6 STR9 Analog Grounding Arrangements

The analog ground (Avss) pin on the STR9 CPU is not connected to the digital ground on the module itself. This connection must be made via solder bridge P504 on the baseboard. By default, this is open so that the Avss is independent. This means that the user must ensure that either P504 is closed or that the Avss pin is not subjected to voltages that will cause internal damage to the STR9.

It is hoped on later revision STR9 CPU Modules that a jumper option between Avss and SGND will be provided on the STR9 module itself. All the other CPU Modules have this feature.

Alternatively for application where the user has an Analogue module fitted it is possible to connect the Avss to the SGND via a zero ohm link on this board also.