Efabless Caravel SoC and user project harness

Memory Mapped I/O summary by address

Address (bytes)	Function
0x00 00 00 00	Flash SPI / overlaid DFFRAM (1k words) start of memory block
0x00 00 0f ff	End of SRAM
0x01 00 00 00	Flash SPI / overlaid SRAM (512 words) start of memory block
0x01 00 07 ff	End of SRAM
0x10 00 00 00	Flash SPI start of program block Program to run starts here on reset.
0x10 ff ff ff	Maximum SPI flash addressable space (16MB) with QSPI 3-byte addressing
0x1f ff ff ff	Maximum SPI flash addressable space (32MB) with 4-byte addressing
0x20 00 00 00	UART clock divider select (system clock freq. / baud rate)
0x20 00 00 04	UART data (returns `0xffffffff` if receiver buffer is empty)
0x20 00 00 08	UART enable
0x21 00 00 00	GPIO input/output (bit 16/bit 0) 1 general-purpose digital, management area only
0x21 00 00 04	GPIO output enable (1 = output, 0 = input)
0x21 00 00 08	GPIO pullup enable (1 = pullup, 0 = none)
0x21 00 00 0c	GPIO pulldown enable (1 = pulldown, 0 = none)
0x22 00 00 00	Counter/Timer 0 configuration register (lower 4 bits) bit 0 = enable (0 = hold, 1 = count) bit 1 = oneshot (0 = continuous count, 1 = one-shot count) bit 2 = updown (0 = count down, 1 = count up) bit 3 = irq enable (0 = disabled, 1 = trigger IRQ channel 10 on timeout)
0x22 00 00 04	Counter/Timer 0 current value Set or read the 32-bit current value.
0x22 00 00 08	Counter/Timer 0 reset value Set or read the 32-bit reset (down-count) or compare (up-count) value.
0x23 00 00 00	Counter/Timer 1 configuration register (lower 4 bits) bit 0 = enable (0 = hold, 1 = count) bit 1 = oneshot (0 = continuous count, 1 = one-shot count) bit 2 = updown (0 = count down, 1 = count up) bit 3 = irq enable (0 = disabled, 1 = trigger IRQ channel 11 on timeout)
0x23 00 00 04	Counter/Timer 1 current value Set or read the 32-bit current value.
0x23 00 00 08	Counter/Timer 1 reset value Set or read the 32-bit reset (down-count) or compare (up-count) value.
0x24 00 00 00	SPI master configuration register bits 0–7 = prescaler (core clock / (prescaler + 1) = SPI clock rate / 2) (default 2) bit 8 = mlb (0 = msb first, 1 = lsb first) (default 0) bit 9 = invcsb (0 = csb active low, 1 = csb active high) (default 0) bit 10 = invsck (0 = normal sck, 1 = inverted sck) (default 0) bit 11 = mode (0 = read/write on opposite sck edge, 1 = same edge) (default 0) bit 12 = stream (0 = raise csb after each byte, 1 = keep csb low until stream bit cleared) bit 13 = enable (0 = SPI master disabled, 1 = SPI master enabled) bit 14 = irq enable (0 = disabled, 1 = SPI read valid triggers interrupt channel 9) bit 15 = housekeeping (0 = disconnected, 1 = connected)
0x24 00 00 04	SPI master data register (low 8 bits) Write data to send to low byte or read received data from low byte.

Memory Mapped I/O summary by address *(continued)*

Address (bytes)	Function	
`0x25 00 00 00`	Logic Analyzer Data 0 (bits 31 to 0)	
`0x25 00 00 04`	Logic Analyzer Data 1 (bits 63 to 32)	
`0x25 00 00 08`	Logic Analyzer Data 2 (bits 91 to 64)	
`0x25 00 00 0c`	Logic Analyzer Data 3 (bits 127 to 92)	
`0x25 00 00 10`	Logic Analyzer Output Enable 0 (sense inverted—OEB)	
`0x25 00 00 14`	Logic Analyzer Output Enable 1 (sense inverted—OEB)	
`0x25 00 00 18`	Logic Analyzer Output Enable 2 (sense inverted—OEB)	
`0x25 00 00 1c`	Logic Analyzer Output Enable 3 (sense inverted—OEB)	
`0x25 00 00 20`	Logic Analyzer Input Enable 0	
`0x25 00 00 24`	Logic Analyzer Input Enable 1	
`0x25 00 00 28`	Logic Analyzer Input Enable 2	
`0x25 00 00 2c`	Logic Analyzer Input Enable 3	
`0x25 00 00 30`	Logic Analyzer Sample	
`0x26 00 00 00`	User project area GPIO data transfer (bit 0, auto-zeroing)	Mirrors housekeeping register 0x13
`0x26 00 00 04`	User project area GPIO power[0] configure	(These are currently undefined/unused.)
		Housekeeping register 0x6e
`0x26 00 00 0c`	User project area GPIO data (L) (GPIO 31 to 0)	Housekeeping registers 0x6a—0x6d
`0x26 00 00 10`	User project area GPIO data (H) (GPIO 37 to 32, upper bits unused)	
		Housekeeping register 0x69
`0x26 00 00 24`	User project area GPIO mprj_io[0] configure	Housekeeping registers 0x1d, 0x1e—
:		:
`0x26 00 00 b8`	User project area GPIO mprj_io[37] configure	Housekeeping registers —0x67, 0x68

bit 0 = management control enable (0 = user control, 1 = management control) (default 1)
bit 1 = output disable (0 = output enabled, 1 = output disabled) (default 1)
bit 2 = hold override value (value = value during hold mode) (default 0)
bit 3 = input disable (0 = input enabled, 1 = input disabled) (default 0)
bit 4 = IB mode select (0 = , 1 =)
bit 5 = analog bus enable (0 = disabled, 1 = enabled)
bit 6 = analog bus select (0 = , 1 =)
bit 7 = analog bus polarity (0 = , 1 =)
bit 8 = slow slew (0 = fast slew, 1 = slow slew) (default 0)
bit 9 = input voltage trip point select (0 = , 1 =)
bits 10–12 = digital mode (see below) (default set in file user_defines.v)

Digital mode bits bit 12 11 10	Digital mode description
0 0 0	analog function (all digital buffers disabled)
0 0 1	input (output disabled), no pullup or pulldown
0 1 0	input (output disabled), with pullup
0 1 1	input (output disabled), with pulldown
1 0 0	open drain (to power) output
1 0 1	open drain (to ground) output
1 1 0	output enabled
1 1 1	weak output (through 5k resistor)

Memory Mapped I/O summary by address *(continued)*

Address (bytes)	Function	
0x26 10 00 00	Housekeeping SPI status (undefined/unused)	Mirrors housekeeping SPI register 0x00
0x26 10 00 04	Chip ID (20 bits, vendor and chip identifiers)	Housekeeping registers 0x01—0x03
0x26 10 00 08	User ID (32 bits, user project and MPW run identifiers)	Housekeeping registers 0x04—0x07
0x26 10 00 0c	DLL enables	Housekeeping register 0x08
0x26 10 00 10	DLL bypass	Housekeeping register 0x09
0x26 10 00 14	Manual interrupt	Housekeeping register 0x0a
0x26 10 00 18	Manual reset (note: this will reset the CPU, not recommended to be called from software.)	Housekeeping register 0x0b
0x26 10 00 1c	DLL manual trim for DCO mode (28 bits)	Housekeeping registers 0x0d—0x10
0x26 10 00 20	DLL output dividers (3 bits each)	Housekeeping register 0x11
0x26 10 00 24	DLL feedback divider	Housekeeping register 0x12
0x26 10 00 28	Trap state (note: when read from the CPU, this value can only be zero by definition.)	Housekeeping register 0x0c
0x26 10 00 2c	SRAM read-only port data	Housekeeping registers 0x16—0x19
0x26 10 00 30	SRAM read-only port address	Housekeeping register 0x15
0x26 10 00 34	SRAM read-only control	Housekeeping register 0x14
0x26 20 00 00	Power good (4 bits: User 1/2 vccd, user 1/2 vdda)	Housekeeping register 0x1a
0x26 20 00 04	Output monitoring (low 3 bits) bit 0 = core clock monitor bit 1 = user clock monitor bit 2 = CPU trap monitor	The PLL clock (crystal oscillator clock multiplied up by PLL) can be viewed on the GPIO pin. The GPIO pin cannot be used as general-purpose I/O when selected for PLL clock output. It is unlikely that a full-speed (100MHz) clock will be able to toggle the GPIO at full swing, but is detectable. The CPU fault state (trap) can be viewed at the GPIO pin as a way to monitor the CPU trap state externally. Housekeeping register 0x1b
0x26 20 00 0c	IRQ input source (low 2 bits) bit 0 = IRQ 7 source bit 1 = IRQ 8 source	The GPIO input can be used as an IRQ event source and passed to the CPU through IRQ channel 7. When used as an IRQ source, the GPIO pin must be configured as an input. Housekeeping register 0x1c
0x26 20 00 10	Housekeeping SPI disable (low bit)	Housekeeping register 0x6f
0x2f 00 00 00	User IRQ enables (low 3 bits)	
0x2f 00 00 04	User wishbone input enable (low bit)	
0x2d 00 00 00	QSPI controller config bit 31 MEMIO enable (reset = 1) 0 = bit-bang mode bit 22 DDR enable bit 21 QSPI enable bit 20 CRM enable bits 19-16 Read latency cycles bits 11-8 I/O output enable bits (bit bang mode) bit 5 Chip select line (bit bang mode) bit 4 Serial clock line (bit bang mode) bits 3-0 Data bits (bit bang mode)	Note: These can only be used if GPIO pins 36 and 37 are configured for management output.*
0x30 00 00 00	User area base	A user project may define additional wishbone slave modules starting at this address.
0x3f ff ff ff	User area top	

* The availability of the QSPI function is dependent on whether or not it is supported by the type of management core SoC implemented.