Housekeeping SPI

SDI (pin F9), CSB (pin E8), SCK (pin F8), and SDO (pin E9)

The “housekeeping” SPI is an SPI slave that can be accessed from a remote host through a standard 4-pin serial interface. The SPI implementation is mode 0, with new data on SDI captured on the SCK rising edge, and output data presented on the falling edge of SCK (to be sampled on the next SCK rising edge). The SPI pins are shared with user area general-purpose I/O.

SPI protocol definition

All input is in groups of 8 bits. Each byte is input msb first.

Every command sequence requires one command word (8 bits) followed by one address word (8 bits) followed by one or more data words (8 bits each), according to the data transfer modes defined below.

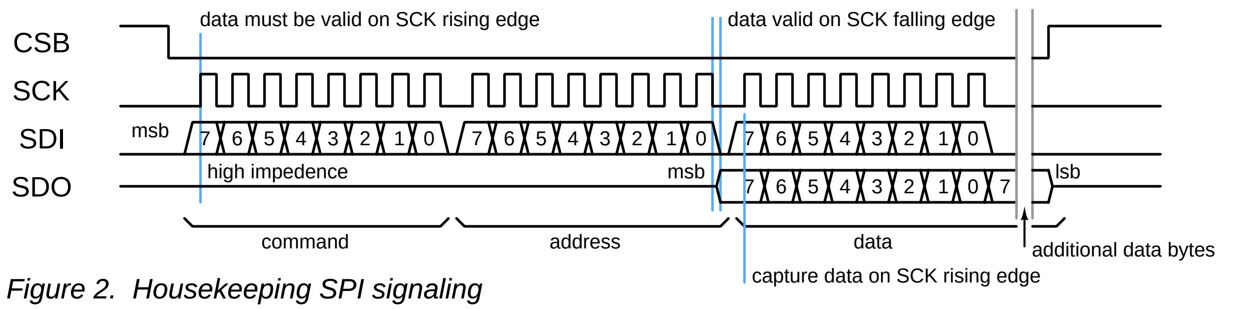
CSB
SCK
SDI
SDO
data must be valid on SCK rising edge
data valid on SCK falling edge
msb
7 6 5 4 3 2 1 0
high impedence
msb
lsb
command
address
data
additional data bytes
capture data on SCK rising edge

Figure 2. Housekeeping SPI signaling

Addresses are read in sequence from lower values to higher values.

Therefore groups of bits larger than 8 should be grouped such that the lowest bits are at the highest address. Any bits additional to an 8-bit boundary should be at the lowest address.

Data are captured from the register map in bytes on the falling edge of the last SCK before a data byte transfer. Multi-byte transfers should ensure that data do not change between byte reads.

CSB pin must be low to enable an SPI transmission. Data are clocked by pin SCK, with data valid on the rising edge of SCK. Output data are received on the SDO line. SDO is held high-impedance when CSB is high and at all times other than the transfer of data bits on a read command. SDO outputs become active on the falling edge of SCK, such that data are written and read on the same SCK rising edge.

After CSB is set low, the SPI is always in the "command" state, awaiting a new command.

The first transferred byte is the command word, interpreted according to Table 8 below.

Table 8 Housekeeping SPI command word definition

`00000000`	No operation
`10000000`	Write in streaming mode
`01000000`	Read in streaming mode
`11000000`	Simultaneous Read/Write in streaming mode
`11000100`	Pass-through (management) Read/Write in streaming mode
`11000110`	Pass-through (user) Read/Write in streaming mode
`10nnn000`	Write in n-byte mode (up to 7 bytes).
`01nnn000`	Read in n-byte mode (up to 7 bytes).
`11nnn000`	Simultaneous Read/Write in n-byte mode (up to 7 bytes).

All other words are reserved and act as no-operation if not defined by the SPI slave module.

SPI protocol definition *(continued)*

The two basic modes of operation are "streaming mode" and "n-byte mode". In "streaming mode" operation, data are sent or received continuously, one byte at a time, with the internal address incrementing for each byte. Streaming mode operation continues until CSB is raised to end the transfer.

In "n-byte mode" operation, the number of bytes to be read and/or written is encoded in the command word, and may have a value from 1 to 7 (note that a value of zero implies streaming mode). After n bytes have been read and/or written, the SPI returns to waiting for the next command. No toggling of CSB is required to end the command or to initiate the following command.

Pass-thru mode

The pass-thru mode puts the CPU into immediate reset, then sets FLASH_CSB low to initiate a data transfer to the QSPI flash. After the pass-thru command byte has been issued, all subsequent SPI signaling on SDI and SCK are applied directly to the QSPI flash (pins FLASH_IO0 and FLASH_CLK, respectively), and the QSPI flash data output (pin FLASH_IO1) is applied directly to SDO, until the CSB pin is raised. When CSB is raised, the FLASH_CSB is also raised, terminating the data transfer to the QSPI flash. The CPU is brought out of reset, and starts executing instructions at the program start address.

This mode allows the QSPI flash to be programmed from the same SPI communication channel as the housekeeping SPI, without the need for additional wiring to the QSPI flash chip.

There are two pass-thru modes. The first one corresponds to the primary SPI flash used by the management SoC. The second one corresponds to a secondary optional SPI flash that can be defined in the user project. The pass-thru mode allows a communications chip external to the Caravel chip program either SPI flash chip from a host computer without requiring separate external access to the SPI flash. Both pass-thru modes only connect to I/O pins 0 and 1 of the SPI flash chips, and so must operate only in the 4-pin SPI mode. The user project may elect to operate the SPI flash in quad mode using a 6-pin interface.

Housekeeping SPI registers

The purpose of the housekeeping SPI is to give access to certain system values and controls independently of the CPU. The housekeeping SPI can be accessed even when the CPU is in full reset. Some control registers in the housekeeping SPI affect the behavior of the CPU in a way that potentially can be detrimental to the CPU operation, such as adjusting the trim value of the digital frequency-locked loop generating the CPU core clock.

Under normal working conditions, the SPI should not need to be accessed unless it is to adjust the clock speed of the CPU. All other functions are purely for test and debug.

The housekeeping SPI can be accessed by the CPU from a running program by enabling the SPI master, and enabling the bit that connects the internal SPI master directly to the housekeeping SPI. This configuration then allows a program to read, for example, the user project ID of the chip. See the SPI master description for details.

manufacturer_ID register address 0x01 low 4 bits and register address 0x02

The 12-bit manufacturer ID for efabless is 0x456

product_ID register address 0x03

The product ID for the Caravel harness chip is 0x10

Housekeeping SPI registers *(continued)*

user project ID register addresses 0x04 to 0x07

The 4-byte (32 bit) user project ID is metal-mask programmed on each project before tapeout, with a unique number given to each user project.

PLL enable register address 0x08 bit 0

This bit enables the digital frequency-locked-loop clock multiplier. The enable should be applied prior to turning off the PLL bypass to allow the PLL time to stabilize before using it to drive the CPU clock.

PLL DCO enable register address 0x08 bit 1

The PLL can be run in DCO mode, in which the feedback loop to the driving clock is removed, and the system operates in free-running mode, driven by the ring oscillator which can be tuned between approximately 90 to 200 MHz by setting the trim bits (see below).

PLL bypass register address 0x09 bit 0

When enabled, the PLL bypass switches the clock source of the CPU from the PLL output to the external CMOS clock (pin C9). The default value is 0x1 (CPU clock source is the external CMOS clock).

CPU IRQ register address 0x0A bit 0

This is a dedicated manual interrupt driving the CPU IRQ channel 6. The bit is not self-resetting, so while the rising edge will trigger an interrupt, the signal must be manually set to zero before it can trigger another interrupt.

CPU reset register address 0x0B bit 0

The CPU reset bit puts the entire CPU into a reset state. This bit is not self-resetting and must be set back to zero manually to clear the reset state.

CPU trap register address 0x0C bit 0

If the CPU has stopped after encountering an error, it will raise the trap signal. The trap signal can be configured to be read from a GPIO pin, but as the GPIO state is potentially unknowable, the housekeeping SPI can be used to determine the true trap state.

PLL trim register addresses 0x0D (all bits) to 0x10 (lower 2 bits)

The 26-bit trim value can adjust the DCO frequency over a factor of about two from the slowest (trim value 0x3ffffff) to the fastest (trim value 0x0). Default value is 0x3ffefff (slow trim, -1). Note that this is a thermometer-code trim, where each bit provides an additional (approximately) 250 ps delay (on top of a fixed delay of 4.67 ns). The fastest output frequency is approximately 215 MHz while the slowest output frequency is approximately 90 MHz.

PLL output divider register address 0x11 bits 2–0

The PLL output can be divided down by an integer divider to provide the core clock frequency. This 3-bit divider can generate a clock divided by 2 to 7. Values 0 and 1 both pass the undivided PLL clock directly to the core (and should not be used, as the processor does not operate at these frequencies).

PLL output divider (2) register address 0x11 bit 5–3

The PLL 90-degree phase output is passed through an independent 3-bit integer clock divider and provided to the user project space as a secondary clock. Values 0 and 1 both pass the undivided PLL clock, while values 2 to 7 pass the clock divided by 2 to 7, respectively.

Housekeeping SPI registers *(continued)*

PLL feedback divider register address 0x12 bits 4–0

The PLL operates by comparing the input clock (pin C9) rate to the rate of the PLL clock divided by the feedback divider value (when running in PLL mode, not DCO mode). The feedback divider must be set such that the external clock rate multiplied by the feedback divider value falls between 90 and 214 MHz (preferably centered on this range, or approximately 150 MHz). For example, when using an 8 MHz external clock, the divider should be set to 19 (19 * 8 = 152). The DCO range and the number of bits of the feedback divider implies that the external clock should be no slower than around 4 to 5 MHz.

Housekeeping SPI register map

Register Address	*msb* 7	6	5	4	3	2	1	*lsb* 0	*comments*
0x00	SPI status and control								unused/ undefined
0x01	unused				manufacturer_ID[11:8] (= 0x4)				read-only
0x02	manufacturer_ID[7:0] (= 0x56)								read-only
0x03	product_ID (= 0x11)								read-only
0x04–0x07	user_project_ID (unique value per project)								read-only
0x08	unused						DLL DCO enable	DLL enable	default 0x02
0x09	unused							DLL bypass	default 0x01
0x0A	unused							CPU IRQ	default 0x00
0x0B	unused							CPU reset	default 0x00
0x0C	unused							CPU trap	read-only
0x0D–0x10	DCO trim (26 bits) (= 0x3ffefff)								default 0x3ffefff
0x11	unused		PLL output divider 2			PLL output divider			default 0x12
0x12	unused			PLL feedback divider					default 0x04
0x13		serial data 2	serial data 1	serial clock	serial load	serial $\overline{\text{reset}}$	serial enable	serial xfer/ busy*	bits 6 to 1 are bit-bang control.

* Bit is write-only for serial transfer, read-only for serial busy. During transfer, the busy bit is one. Transfer is complete when the busy bit returns to zero.

Housekeeping SPI register map, continued

Register Address	*msb* 7	6	5	4	3	2	1	*lsb* 0	*comments*
0x14	unused						SRAM r/o clock	SRAM r/o csb	default 0x00
0x15	SRAM read-only address								default 0x00
0x16–0x19	SRAM read-only data (32 bits)								read-only
0x1a	unused				user1 vccd good	user2 vccd good	user1 vdda good	user2 vdda good	read-only
0x1b	unused					core clock monitor	user clock monitor	trap state monitor	default 0x00
0x1c	unused						irq 2 source	irq 1 source	default 0x00
0x1d–0x68	GPIO configuration data staging (see table at bottom)								
0x69	unused		GPIO data for mprj_io[37:32]						default 0x00
0x6a–0x6d	GPIO data for mprj_io[31:0]								default 0x00
0x6e	unused				user domain power control				unused/ undefined
0x6f	unused							house-keeping disable	default 0x00

GPIO configuration bits (2 bytes per GPIO, using GPIO mprj_io[0] as an example)

0x1d	unused			DM[2]	DM[1]	DM[0]	trip point	slow slew	
0x1e	analog polarity	analog select	analog enable	IB mode select	input disable	hold value	output enable	mgmt enable	