DIF Command Interface

Version 1.3 (5.11.08)

The DIF developers

0 Introduction

A simplified block diagram of the communication between DIF and LDA, or as it is proposed to a PC via USB bus, is shown in Fig. 1. Only the interface between LDA and DIF is shown, the remaining functionalities are combined on DIF- and LDA-side in the blocks “Command Coder/Decoder”. The USB-DIF link is generally for debugging, but as the more simple interface, it will be the first system realization stage as well. Data transfer between LDA and the slabs (front-end ASICs) might happen in two steps (by two commands from the LDA), for example the slow-control data: 1. Load slow-control data from LDA to DIF (data is stored in DIF memory). 2. Transfer data from the memory to the slabs (ASICs).

0.1 LDA-DIF interface

The only connection between LDA and DIF is realized with a 19-pin HDMI cable. Commands are in general 8b/10b-channel coded, while the coding and decoding blocks are fully transparent for the remaining logic/electronics. The DIF is operated with the clock from the LDA: no PLL (DCM) on the DIF, by which a fully synchronous operation of all the DIFs that are connected to one LDA is guaranteed. Clock speed (default): 80MHz (but also possible: 40-120MHz). Fast Commands from the LDA to the DIF like a trigger are transported without channel coding, as well as fast commands from DIF to the LDA like the signal “RAMFull”.

0.2 PC (USB) to DIF interface

The USB interface should “emulate” the LDA-DIF interface as much as possible in order to allow an easy switching to the LDA-DIF setup. The USB-interface does not use the 8b/10b channel coders/decoders. The DIF clock may be generated from the USB side or by a local oscillator. In the USB-setup, PLLs (DCMs) within the DIF FPGA are allowed. On the PC, Labview is a possible operating system, but not mandatory.

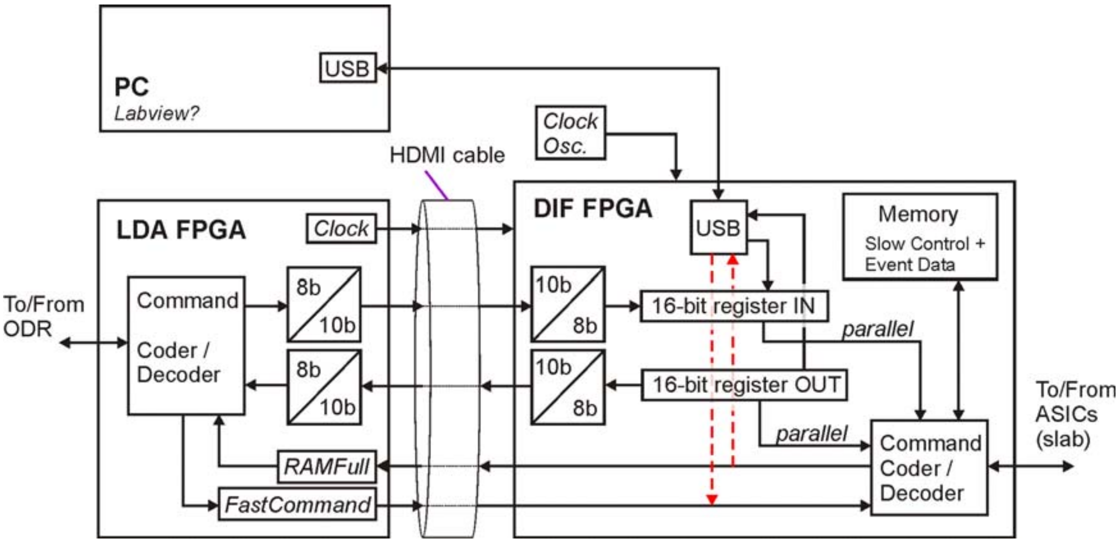


Figure 1: Interface of the DIF FPGA to the LDA, or a PC via USB

1 Transfer between LDA and DIF

The data transfer between LDA and DIF is 8b/10b channel coded. The 8b/10b coding is realized by a 5b/6b and a 3b/4b coder. On the 8-bit side, a respective komma character (K) is referenced to by K*X.Y*, while *X* is the 5-bit word, and *Y* is the 3-bit word, both in decimal notation. For example K28.1 is the 8-bit sequence 11100 001.

Two types of data transfer “frames” are defined between LDA and DIF [1, 2]:

1.1 Command Frame

A command frame is 16-bit long:

15 8	7 0
komma character (K)	command word (D)

The komma character K and the command word D are referenced to by K*X.Y* and D*X.Y*, respectively: X has a **5-bit resolution**, Y has a **3-bit resolution**. E.g. K28.1 is the 8-bit sequence 11100 001.

1.2 Block Transfer

A block transfer is used to transmit configuration-, result- or status information data between LDA and DIF. The length is not fixed, although only an even number of 16-bit words is allowed (see K23.7 in Table 1 and /EPD/ in Table 2). A block transfer is always framed by command frames with the komma characters K27.7 and K29.7, respectively (see section 1.3).

total block length	start address	data	CRC
16 bit	16 bit	n * 16 bits	16 bit

1.3 Komma Characters (see section 1.1) and special sequences [1]

The 8b/10b channel coding allows for channel synchronization and maintenance the so called komma characters (K):

Komma Character K	Task (Meaning)
K28.0	Signals the next symbol (command word) is a SYNCCMD.
K28.1	
K28.2	
K28.3	Signals the next symbol (command word) is a COMMAND.
K28.4	
K28.5	reserved for link synchronization
K28.6	
K28.7	reserved for link synchronization
K23.7	Carrier Extend. Used to PAD the end of a data frame out to an even number of Symbols, so that next frame, or IDLE sequence starts on an even footing. **/R/**
K27.7	Start of data frame (block transfer) **/S/**
K29.7	End of data frame (block transfer) **/T/**
K30.7	

Table 1: Komma Characters

Several special sequences are defined:

Set	Sequence	Comment
/I1/	/K28.5/D5.6/	Idle sequence, sent when running DP is +, flips it to -. Sent automatically.
/I2/	/K28.5/D16.2/	Idle sequence, sent when running DP is -, maintains it as -. Sent automatically.
/EPD/	**/T/R/** or **/T/R/R/**	Used to end a data frame, the addition of an extra /R/ is used to pad things out to an even number.
/LOOP/	/K28.5/D12.6/	Low-level link loop back start (DON'T SEND from User-Logic)
/ENDLOOP/	/K28.5/D16.7/	Low-level link loop back end (DON'T SEND from User-Logic)
/LINKSTART/	/K28.5/D1.4/	Link Start. (DON'T SEND from User-Logic)
/LINKACK/	/K28.5/D30.3/	Link Start ACK. (DON'T SEND from User-Logic)

Table 2: Special Sequences

2 DIF Commands (from LDA or USB to DIF FPGA)

The following commands/command registers are defined (see detailed explanation of the command registers afterwards):

Command	komma character	command word D	Operation	Change DIF State?
power_on	K28.3	D1.1 D1.2 D1.3	turn power regulators on turn power regulators off read power register	no
reset	K28.3	D2.1 D2.2 D2.3 D2.4 D2.5	reset of DIF reset of slab reset all reset BCID read reset register	yes
set_DIF_mode	K28.3 K28.3 K28.0 K28.0 K28.3	D3.1 D3.2 D3.3 D3.4 D3.6	set detector into “SLEEP” set detector into “IDLE” set DIF into sync mode put DIF into loopback mode read DIF_mode register	yes
power_pulsing	K28.3	D4.1 D4.2 D4.3 D4.4 D4.5 D4.6 D4.7	switch pwr_analog switch pwr_digital switch pwr_ss/pwr_sca switch pwr_adc switch pwr_dac switch all on/off read power_pulsing register	no
load_sc_data	K28.3	D5.1 D5.2 D5.3 D5.4	load data from LDA to DIF load data from DIF to slab readback current data read load_sc_data reg.	yes
start_acquire	K28.3	D6.1 D6.2 D6.3 D6.4 D6.5	start data-taking (int. trig) start data-taking (ext. trigger) stop data-taking read data slab to DIF read data DIF to LDA	yes

Command	komma character	command word D	Operation	Change DIF State?
		D6.6	read start_acquire reg.	
control_status_regs	K28.3	D7.1	readout of DIF status register	no
		D7.2	readout of DIF control reg.	
		D7.3	set DIF control register (block transfer follows)	
readout_info	K28.3	D9.1	read DIF board-ID	no
		D9.2	read board's production date	
		D9.3	read FPGA firmware version	
		D9.4	read firmware date	
			(each: block transfer follows)	
dif_dif_link	K28.3	D12.1	use DIF-DIF connector	no
		D12.2	do not use DIF-DIF link	
		D12.3	read dif_dif_link register	
load_FPGA_firmware	K28.3	D13.1	Load new firmware to FPGA	no
		D13.2	reset FPGA	
#### ECAL specific ####				
		D17.0		
#### DHCAL specific ####				
		D22.0		
#### AHCAL specific ####				
read_temp_power	K28.3		temp/power sensor readout:	no
		D27.1	read temp. results µC to DIF	
		D27.2	temp. results DIF to LDA	
		D27.3	read volt./currents µC to DIF	
		D27.4	volt./currents DIF to LDA	
		D27.5	read temp_power register	
calibrate	K28.3		do a calibration run:	yes
		D28.1	with light sys., int. trig	
		D28.2	with charge sys., int. trig	
		D28.3	with light sys., ext. trig.	
		D28.4	with charge sys., ext. trig.	
		D28.5	read calibrate register	
load_calib_analog	K28.3		Set DACs (pulse heights) and delay regs. of the calibration system:	no
		D29.1	block transfer follows	
		D29.2	Read current settings	
		D29.3	read calib_analog register	
readout_probe_regs	K28.3	D30.1	Read ASICs probe regs	yes
		D30.2	read probe_regs register	
load_uC_software	K28.3	D31.1	load new µC software	yes
		D31.2	reset of µC	
		D31.3	read uC software reg.	

Table 3: Commands from LDA to DIF

ECAL specific (D17.0 – D21.7)
DHCAL specific (D22.0 – D26.7)
AHCAL specific (D27.0 – D31.7)

Change DIF State (last column) means: If the DIF changes its state from „IDLE“ to any other state by a received command, the respective operation should not be interrupted by following commands except for emergencies or resets. After completion of the tasks, the DIF changes back to “IDLE” automatically and is ready for new commands.

2.1 Command Register Description

For each command that is sent from LDA to DIF, the DIF has a dedicated **command register**. The address of this **command register** is defined by the X in the incoming DX.Y command word (see section 1.1). **Command registers** are 16-bit, and can be subdivided for several functional purposes.

The general notation is:

15 10	9 5	4	3	2	1	0
Reserved	Status Bits(4:0)	Bit	Bit	Bit	Bit	Bit
R, +0	RC, +10100	RW, +0	RS, +0	RW, +0	RW, +0	RW, +0

Note: R = Readable by the LDA,
W = Writeable by the LDA,
C = Clearable by the LDA,
S = Settable by the LDA,

+x = Value undefined after reset,
+0 = Value is 0 after reset,
+1 = Value is 1 after reset,

2.1.1 power_on register (command and address D1.Y)

15 1	0
reserved	slab_power
R, +0	RW, +0

Bit no.	Bit Field	Description
15 – 1	reserved	reserved
0	slab power	Switch on or off slab power: slab_power = ‘1’: slab power is on (set by D1.1) slab_power = ‘0’: slab power is off (reset by D1.2)

Table 4: power_on register description

2.1.2 reset register (command and address D2.Y)

15 4	3	2	1	0
reserved	reset_BCID	reset_all	reset_slab	reset_DIF
R, +0	RS, +0	RS, +0	RS, +0	RS, +0

Bit no.	Bit Field	Description
15 – 4	reserved	reserved
3	reset_BCID	Reset bunch counters of the ASICs (on slab, must be synchronous for all ASICs): reset_BCID = ‘1’: reset is active for 4 clock cycles, afterwards the DIF resets this bit automatically (set by D2.4) reset_BCID = ‘0’: reset is not active.
2	reset_all	general reset of DIF and slab electronics: reset_all = ‘1’: reset active for 6 clock cycles, afterwards the DIF resets this bit automatically (set by D2.3) reset_all = ‘0’: reset not active.
1	reset_slab	reset of slab electronics: reset_slab = ‘1’: reset active for 6 clock cycles, after accepting the DIF resets this bit automatically (set by D2.2) reset_slab = ‘0’: reset not active.
0	reset_DIF	reset of DIF electronics: reset_DIF = ‘1’: reset active for 6 clock cycles, after accepting the DIF resets this bit automatically (set by D2.1) reset_DIF = ‘0’: reset not active.

Table 5: Reset register description

2.1.3 DIF_mode register (command and address D3.Y)

15 7	6 4	3	2	1	0
Reserved	current_mode(2:0)	LOOP	SYNC	IDLE	SLEEP
R, +0	R, +000	RS, +0	RS, +0	RS, +0	RS, +0

Bit no.	Bit Field	Description
15 – 7	reserved	reserved

6 - 4	current_mode	Shows the actual mode, the DIF is in, defined by the last “DIF_mode” command from the LDA (read-only):
		current_mode = ‘000’ DIF is in SLEEP mode,
		current_mode = ‘001’ DIF is in IDLE mode,
		current_mode = ‘010’ DIF is in SYNC mode,
		current_mode = ‘011’ DIF is in LOOP mode

3	LOOP	puts DIF into LOOP mode (debugging) LOOP = ‘1’ : DIF is in LOOP mode with LDA, after accepting, the DIF resets this bit automatically and ‘current_mode is set to ‘011’ (set by D3.4) LOOP = ‘0’ : no mode change

Bit no.	Bit Field	Description
2	SYNC	puts DIF into SYNC mode (DIF synchronization) SYNC = ‘1’ : DIF is in SYNC mode, after accepting, the DIF resets this bit automatically and ‘current_mode is set to ‘010’ (set by D3.3) SYNC = ‘0’ : no mode change
1	IDLE	puts DIF into IDLE mode (general wait and ready state): IDLE = ‘1’ : DIF is in IDLE mode, after accepting, the DIF resets this bit automatically and ‘current_mode is set to ‘001’ (set by D3.2) IDLE = ‘0’ : no mode change
0	SLEEP	puts DIF into SLEEP mode (powered-down wait state) SLEEP = ‘1’ : DIF is in SLEEP mode with LDA, after accepting, the DIF resets this bit automatically and ‘current_mode is set to ‘000’ (set by D3.1) SLEEP = ‘0’ : no mode change

Table 6: DIF_mode register description

2.1.4 power_pulsing register (command and address D4.Y)

This command is for debugging only. The power pulsing control should be done automatically by the DIF in order to guarantee a timing-precise switching. E.g., on a “start_acquire”-command (D6.1, D6.2) from the LDA, the DIF switches-on the slab before starting the data-taking.

15 6	5	4	3	2	1	0
Reserved	SLAB	DAC	ADC	SS_SCA	DIGITAL	ANALOG
R, +0	RW, +0	RW, +0	RW, +0	RW, +0	RW, +0	RW, +0

Bit no.	Bit Field	Description
15 – 6	reserved	reserved

5	SLAB	Switch the power of the complete slab, namely the power-pulsing control signals: pwr_analog, pwr_digital, pwr_ss/pwr_sca, pwr_adc, pwr_dac.
		The state of SLAB is toggled by D4.6:
		SLAB = ‘1’ SLAB is switched ON
		SLAB = ‘0’ SLAB is switched OFF

4	DAC	The state of DAC is toggled by D4.5:
		DAC = ‘1’ pwr_dac is switched ON
		DAC = ‘0’ pwr_dac is switched OFF

3	ADC	The state of ADC is toggled by D4.4:
		ADC = ‘1’ pwr_adc is switched ON
		ADC = ‘0’ pwr_adc is switched OFF

2	SS_SCA	The state of SS_SCA is toggled by D4.3:
		SS_SCA = ‘1’ pwr_ss/pwr_sca is switched ON
		SS_SCA = ‘0’ pwr_ss/pwr_sca is switched OFF

1	DIGITAL	The state of DIGITAL is toggled by D4.2:
		DIGITAL = ‘1’ pwr_digital is switched ON
		DIGITAL = ‘0’ pwr_digital is switched OFF

0	ANALOG	The state of ANALOG is toggled by D4.1: ANALOG = ‘1’ pwr_analog is switched ON

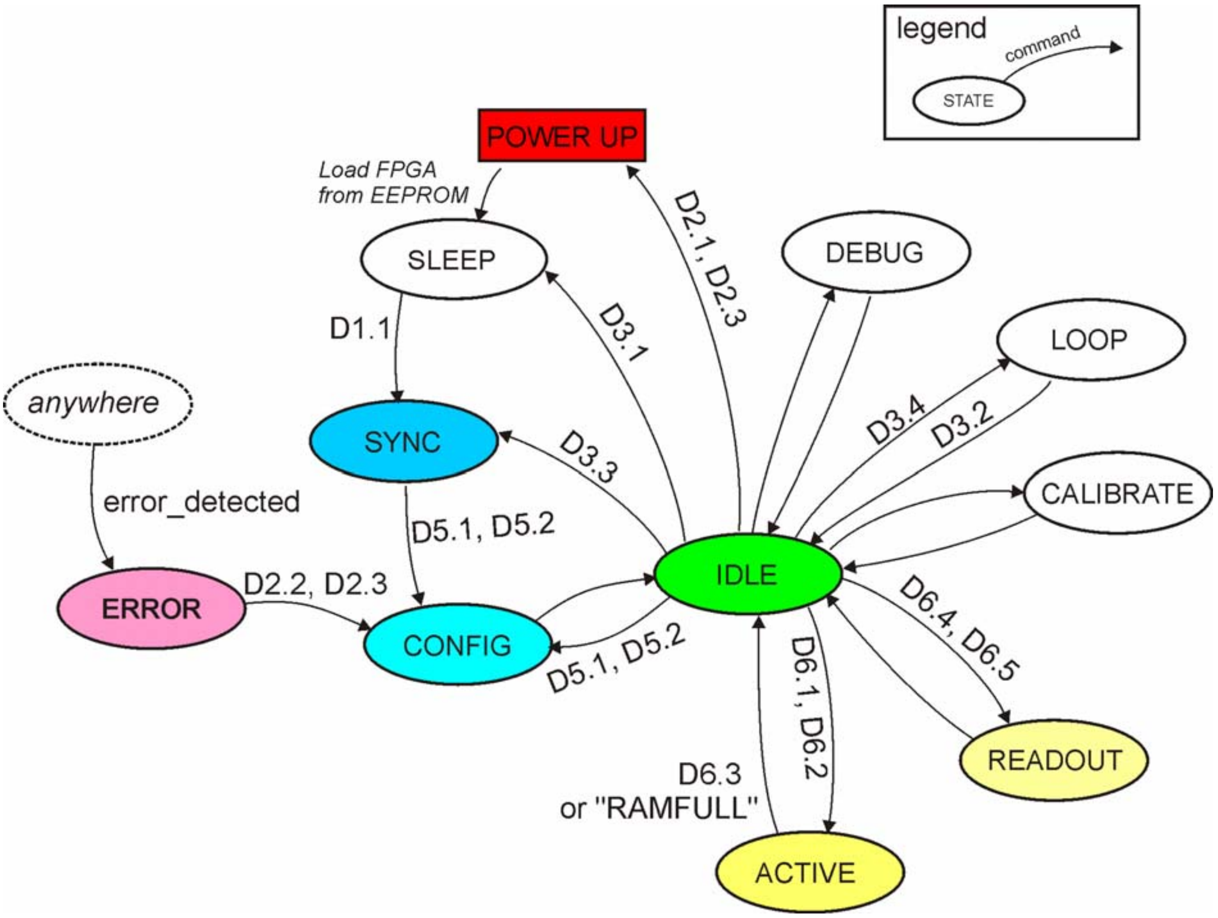
Bit no.	Bit Field	Description
		ANALOG = ‘0’ pwr_analog is switched OFF

Table 7: power_pulsing register description

3 DIF States and State Diagram

DIF States	command that switches DIF into this state	remark
SLEEP	D3.1	powered down wait state
IDLE	D3.2, D6.3	after completion of tasks DIF falls into IDLE state automatically (DIF power on, slab power on minimum)
SYNC	SYNC: D3.3	synchronization between LDA and all the DIFs
CONFIG	D5.1, D5.2	configuration data to ASICs
ACTIVE	D6.1, D6.2	take data (internal or external trigger)
READOUT	D6.4, D6.5	Readout of results from ASICs
LOOP	D3.4	LDA Debug Mode
CONTR/DEBUG	D7.1, D7.2, D7.3, D30.1	read/set DIF status and control registers, readout of ASIC’s probe registers for debugging
CALIBRATE	D28.1, D28.2, D28.3, D28.4	AHCAL: Calibration runs
ERROR	error detected by DIF during a regular data/signal quality check	errors most likely due to configuration losses of slab (ASICs) => new config necessary
“POWER-UP”	D2.1, D2.3	not really a state

Table 8: DIF states



References

[1] Marc Kelly’s web page: http://www.hep.manchester.ac.uk/u/mpkelly/calice/lda/Calice_LDA_Overview.html

[2] Matt Warren et al. “DAQ Status and Overview”, CALICE week Manchester, Electronics Readout session II, Sept. 8th-10th, 2008