**APB SoC controller**

**APB SoC Controller:-**

* **This APB peripheral primarily controls I/O configuration and I/O function connection. It also supports a few registers for miscellaneous functions.**

## **APB SoC CTRL CSRs**

### **INFO offset = 0x0000**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **N\_CORES** | **31:16** | **RO** |  | **Number of cores in design** |
| **N\_CLUSTERS** | **15:0** | **RO** |  | **Number of clusters in design** |

### 

### **BUILD\_DATE offset = 0x000C**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **YEAR** | **31:16** | **RO** |  | **Year in BCD** |
| **MONTH** | **15:8** | **RO** |  | **Month in BCD** |
| **DAY** | **7:0** | **RO** |  | **Day in BCD** |

### 

### **BUILD\_TIME offset = 0x0010**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **HOUR** | **23:16** | **RO** |  | **Hour in BCD** |
| **MINUTES** | **15:8** | **RO** |  | **Minutes in BCD** |
| **SECONDS** | **7:0** | **RO** |  | **Seconds in BCD** |

### 

### **JTAGREG offset = 0x0074**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **TBD** | **31:0** | **R/W** |  | **To Be Determined** |

### 

### **BOOTSEL offset = 0x00C4**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **BootDev** | **0:0** |  |  | **Selects Boot device 1=SPI, 0=Host mode via I2Cs** |

### 

### **CLKSEL offset = 0x00C8**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **S** | **0:0** | **R/W** |  | **This register contains whether the system clock is coming from**  **the FLL or the FLL is bypassed. It is a read-only register by the core but it can be**  **written via JTAG.** |

### 

### **WD\_COUNT offset = 0x00D0**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **COUNT** | **31:0** | **R/W** | **0x8000** | **Only writable before Watchdog is enabled** |

### 

### **WD\_CONTROL offset = 0x00D4**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **ENABLE\_STATUS** | **31:31** | **R0** |  | **1=Watchdog Enabled,** |
|  | | | | **0=Watchdog not enabled.** |
|  | | | | **Note: once enabled, cannot be disabled** |
| **WD\_VALUE** | **15:0** | **WO** | **NA** | **Set to 0x6699 to reset watchdog when enabled, read current WD value** |

### 

### **RESET\_REASON offset = 0x00D8**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **REASON** | **1:0** | **R/W** |  | **2’b01= reset pin, 2’b11=Watchdog expired** |

### 

### **RTO\_PERIPHERAL\_ERROR offset = 0x00E0**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **FCB\_RTO** | **8:8** | **R/W** | **0x0** | **1 indicates that the FCB interface caused a ready timeout** |
| **TIMER\_RTO** | **7:7** | **R/W** | **0x0** | **1 indicates that the TIMER interface caused a ready timeout** |
| **I2CS\_RTO** | **6:6** | **R/W** | **0x0** | **1 indicates that the I2CS interface caused a ready timeout** |
| **EVENT\_GEN\_RTO** | **5:5** | **R/W** | **0x0** | **1 indicates that the EVENT GENERATOR interface caused a ready timeout** |
| **ADV\_TIMER\_RTO** | **4:4** | **R/W** | **0x0** | **1 indicates that the ADVANCED TIMER interface caused a ready timeout** |
| **SOC\_CONTROL\_RTO** | **3:3** | **R/W** | **0x0** | **1 indicates that the SOC CONTROL interface caused a ready timeout** |
| **UDMA\_RTO** | **2:2** | **R/W** | **0x0** | **1 indicates that the UDMA CONTROL interface caused a ready timeout** |
| **GPIO\_RTO** | **1:1** | **R/W** | **0x0** | **1 indicates that the GPIO interface caused a ready timeout** |
| **FLL\_RTO** | **0:0** | **R/W** | **0x0** | **1 indicates that the FLL interface caused a ready timeout** |

### 

### **READY\_TIMEOUT\_COUNT offset = 0x00E4**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **COUNT** | **19:0** | **R/W** | **0xFF** | **Number of APB clocks before a ready timeout occurs** |

### 

### **RESET\_TYPE1\_EFPGA offset = 0x00E8**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **RESET\_LB** | **3:3** | **R/W** | **0x0** | **Reset eFPGA Left Bottom Quadrant** |
| **RESET\_RB** | **2:2** | **R/W** | **0x0** | **Reset eFPGA Right Bottom Quadrant** |
| **RESET\_RT** | **1:1** | **R/W** | **0x0** | **Reset eFPGA Right Top Quadrant** |
| **RESET\_LT** | **0:0** | **R/W** | **0x0** | **Reset eFPGA Left Top Quadrant** |

### 

### **ENABLE\_IN\_OUT\_EFPGA offset = 0x00EC**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **ENABLE\_EVENTS** | **5:5** | **R/W** | **0x00** | **Enable events from efpga to SOC** |
| **ENABLE\_SOC\_ACCESS** | **4:4** | **R/W** | **0x0** | **Enable SOC memory mapped access to EFPGA** |
| **ENABLE\_TCDM\_P3** | **3:3** | **R/W** | **0x0** | **Enable EFPGA access via TCDM port 3** |
| **ENABLE\_TCDM\_P2** | **2:2** | **R/W** | **1x0** | **Enable EFPGA access via TCDM port 2** |
| **ENABLE\_TCDM\_P1** | **1:1** | **R/W** | **2x0** | **Enable EFPGA access via TCDM port 1** |
| **ENABLE\_TCDM\_P0** | **0:0** | **R/W** | **3x0** | **Enable EFPGA access via TCDM port 0** |

### **EFPGA\_CONTROL\_IN offset = 0x00F0**

| Field | Bits | Access | Default | Description |
| --- | --- | --- | --- | --- |
| EFPGA\_CONTROL\_IN | 31:0 | R/W | 0x00 | EFPGA control bits use per eFPGA design |

### **EFPGA\_STATUS\_OUT offset = 0x00F4**

| Field | Bits | Access | Default | Description |
| --- | --- | --- | --- | --- |
| EFPGA\_STATUS\_OUT | 31:0 | RO |  | Status from eFPGA |

### **EFPGA\_VERSION offset = 0x00F8**

| Field | Bits | Access | Default | Description |
| --- | --- | --- | --- | --- |
| EFPGA\_VERSION | 7:0 | RO |  | EFPGA version info |

### **SOFT\_RESET offset = 0x00FC**

| Field | Bits | Access | Default | Description |
| --- | --- | --- | --- | --- |
| SOFT\_RESET | 1:1 | WO |  | Write only strobe to reset all APB clients |

### **IO\_CTRL offset = 0x0400**

**I/O control supports two functions:**

* **I/O configuration**
* **I/O function selection**

**I/O configuration (CFG) is a series of bits that may be used to control I/O PAD characteristics, such as drive strength and slew rate. These driver control characteristics are implementation technology dependent and are TBD. I/O selection (MUX) controls the select field of a mux that connects the I/O to different signals in the device.**

**Each port is individually addressable at offset + IO\_PORT \* 4. For example, the IO\_CTRL CSR for IO\_PORT 8 is at offset 0x0420.**

| **Field** | **Bits** | **Type** | **Default** | **Description** |
| --- | --- | --- | --- | --- |
| **CFG** | **13:8** | **RW** | **0x00** | **Pad configuration (TBD)** |
| **MUX** | **1:0** | **RW** | **0x00** | **Mux select** |

**Theory of Operation:-**

* **Ports:-**
  + **input logic HCLK,**
  + **input logic HRESETn,**
  + **input ref\_clk\_i,**
  + **input rstpin\_ni,**
  + **input logic [APB\_ADDR\_WIDTH-1:0] PADDR,**
  + **input logic [ 31:0] PWDATA,**
  + **input logic PWRITE,**
  + **input logic PSEL,**
  + **input logic PENABLE,**
  + **output logic [ 31:0] PRDATA,**
  + **output logic PREADY,**
  + **output logic PSLVERR,**
  + **input logic sel\_fll\_clk\_i,**
  + **input logic bootsel\_i,**
  + **input [31:0] status\_out,**
  + **input [ 7:0] version,**
  + **input stoptimer\_i,**
  + **input dmactive\_i,**
  + **output logic wd\_expired\_o,**
  + **output logic [31:0] control\_in,**
  + **output logic [`N\_IO-1:0][`NBIT\_PADCFG-1:0] pad\_cfg\_o,**
  + **output logic [`N\_IO-1:0][`NBIT\_PADMUX-1:0] pad\_mux\_o,**
  + **input logic [JTAG\_REG\_SIZE-1:0] soc\_jtag\_reg\_i,**
  + **output logic [JTAG\_REG\_SIZE-1:0] soc\_jtag\_reg\_o,**
  + **output logic [31:0] fc\_bootaddr\_o,**
  + **// eFPGA connections**
  + **output logic clk\_gating\_dc\_fifo\_o,**
  + **output logic [3:0] reset\_type1\_efpga\_o,**
  + **output logic enable\_udma\_efpga\_o,**
  + **output logic enable\_events\_efpga\_o,**
  + **output logic enable\_apb\_efpga\_o,**
  + **output logic enable\_tcdm3\_efpga\_o,**
  + **output logic enable\_tcdm2\_efpga\_o,**
  + **output logic enable\_tcdm1\_efpga\_o,**
  + **output logic enable\_tcdm0\_efpga\_o,**
  + **output logic fc\_fetchen\_o,**
  + **output logic rto\_o,**
  + **input logic start\_rto\_i,**
  + **input logic [`NB\_MASTER-1:0] peripheral\_rto\_i,**
  + **output logic soft\_reset\_o**

* **In reset mode that is when HRESETn is made low,register is set to the default values.The watchdog timer is disabled,watchdog counter is set to default value 32768,state if the module is set to IDLE.Assign fc\_bootaddr\_o to 32'h1A000080; fc\_fetchen\_o, pad\_cfg\_o,clk\_gating\_dc\_fifo set to 1. READY\_TIMEOUT\_COUNT is set to 20’h000ff.Remaining all the outputs are low.**
* **At every positive edge of the clock HCLKn,** 
  + **If the watchdog timer is in reset mode,in this clock edge the reset mode is turned off.**
  + **If start\_rto\_i is high then the ready\_timeout\_count will be decrementing ,else if start\_rto\_i is low,then the ready\_timeout\_count is set to the default value in register READY\_TIMEOUT\_COUNT .**
  + **Whenever ready\_timeout\_count reaches zero then rto\_o is made high.**
  + **Based on the input peripheral\_rto\_i,The register RTO\_PERIPHERAL\_ERROR is updated .**
  + **The output soc\_jtag\_reg\_o changes with right shift and current value of soc\_jtag\_reg\_i will be inserted on MSB side.**
  + **If the module is in WAIT state then it is changed to IDLE state .If PADDR[11:0] is the address of RESET\_REASON then the register value is set to default 0 meaning the reset clear has been commanded.**
  + **If the module is in IDLE state then if PSEL,PENABLE and PWRITE are high then it changes to WRITE state,else if PWRITE is low,then it is in READ state.**
  + **If the state is WRITE state,then PREADY is made high and state is changed to WAIT and operation based on PADDR[11:0] happens.**

**If PADDR[11:0] is:-**

### **WD\_COUNT offset = 0x00D0**

* + - * **The start count of the watchdog timer is changed if the watchdog is enabled otherwise not changed.( PWDATA[30:0]).**

### **WD\_CONTROL offset = 0x00D4**

* + - * **If PWDATA[31] is high ,then watchdog is enabled and reset.**
      * **If the watchdog is already enabled and PWDATA[15:0]=16'h6699,then watchdog is reset.**

### **RTO\_PERIPHERAL\_ERROR offset = 0x00E0**

* + - * **The register RTO\_PERIPHERAL\_ERROR is set to 0.**

### **READY\_TIMEOUT\_COUNT offset = 0x00E4**

* + - * **The register READY\_TIMEOUT\_COUNT is set to {PWDATA[19:4], 4'hf}.**

### **RESET\_TYPE1\_EFPGA offset = 0x00E8**

* + - * **The register RESET\_TYPE1\_EFPGA is set to PWDATA[3:0]**

### **ENABLE\_IN\_OUT\_EFPGA offset = 0x00EC**

* + - * **The register ENABLE\_IN\_OUT\_EFPGA is set to PWDATA[5:0]**

### **EFPGA\_CONTROL\_IN offset = 0x00F0**

* + - * **The register EFPGA\_CONTROL\_IN is set to PWDATA**

### **SOFT\_RESET offset = 0x00FC**

* + - * **All the registers are set to default values.**

### **IO\_CTRL CSRs offset = 0x04??**

* + - * **If PADDR[9:2] is less than the number of ports then, pad\_cfg\_o[PADDR[9:2]] <= PWDATA[8+:`NBIT\_PADCFG] and**

**pad\_mux\_o[PADDR[9:2]]=PWDATA[0+:`NBIT\_PADMUX]**

* + - * **The port number is also stored in a variable.**
      * **Here NBIT\_PADCFG is 6 and NBIT\_PADMUX is 2.**
  + **If the state is in READ mode,then**
    - **Based on the PADDR[11:0] ,corresponding registers are read and sent on the PRDATA.**
    - **PREADY is made high and state is made to WAIT.**
    - **If PWDATA[11:0] is**

### **INFO offset = 0x0000**

* + - * + **INFO register is read**

### **BUILD\_DATE offset = 0x000C**

* + - * + **BUILD\_DATE Register is read into the PRDATA.**
    - **Like above all the registers are read based on the address provided.Here PRADATA is a 32 bit bus.So,if the register width is less than 32 bits,remaining bits till the MSB in PRDATA are filled with 0.**
    - **If PADDR[11:0] is not equal to any of the addresses of CSRs then PSLVERR is made high and PRDATA is written 32'h0095BEEF.**
    - **If the Address PADDR[11:10] == 2'b01 which means the address is like 12’h4?? ,this means we are accessing I/O port configs.**
      * **Then if PADDR[9:2](port number) is less than N\_IO(number of I/O ports) ,then PRDATA[8+:`NBIT\_PADCFG] will store the configuration value of the port and PRDATA[0+:`NBIT\_PADMUX] will store the mux value of the port.**
      * **If PADDR[9:2] is greater than N\_IO then PRDATA is 32'h0095BEEF.**
* **WORKING OF THE WATCHDOG TIMER**
  + **The watchdog timer is sensitive to , and rstpin\_ni.**
  + **When rstpin\_ni is low,then it is set to default where the current wd timer value is set to 32768.**
  + **In active mode ,at every positive edge of the ref\_clk\_i,if watchdog is in reset mode then it resets with current count to the register WD\_COUNT value and the reset mode is turned off in the next HCLK positive edge.**
  + **If it is not in reset mode and is in normal working mode,if watchdog is enabled and stoptimer\_i is low,then current watchdog count is decremented by 1 .Now,If current count reaches 1 then wd\_expired\_o is made high.**
* **WORKING OF RESET\_REASON register**
  + **If rstpin\_ni is low then the register value is 1**
  + **If wd\_expired\_o is high then register value is 2**
  + **If the Reset reason is commanded to be cleared then the register value is made 0.**
* **If HRESET\_n is low then the register BOOTSEL value becomes equal to {dmactive\_i, bootsel\_i}.Otherwise at every positive clock edge it remains the same.**