UP2 EVT3 MAX10

FPGA Spec v0.4

CUSTOMER : xxx

PROJECT: UP3-board / EVT

Author/Dept: Luigi GRASSO (AFI-AAEON EUROPE – akka: LGR)

Date: 2017-03-03

AAEON RD Dept: xxx

AAEON doc dir: xxxx

Document status: p (preliminary)

Document Security level: ACCESSIBLE BY ATI and EMUTEX ONLY

Table of Contents

V0.1 = initial version

V0.2 = updated version for EVT2 test FW

V0.3 = updated conf_reg definition on §2.1

V0.4 = updated fw for EVT3

1 Introduction

This document provides the description of the EVT3/ MAX10 FPGA function, please refer to:

1. "UP-APL01_A03_20170104A.PDF" for schematic and signal naming convention
2. "170222 UP2 MAX10 ref mapping EVT3 v1.7.xlsx" for FPGA mapping
3. "170116 up2max v0.2.0" for RTL code
4. fw MAX10-2K: 170116 up2max(2K) v0.2.0 (cksum 0068B781).pof
5. fw MAX10-4K: 170116 up2max(4K) v0.2.0 (cksum 026B198F).pof

2 FPGA description

The MAX10 FPGA (10M02(04)SC-U169) will allow several functions such as:

1. Rise voltage of HAT-pins from APL-native 1V8 to 3V3 LVTTL required by HAT standard
2. Provide same “bread-board friendly” interface as RPi2 which are fundamental for makers applications : programmable input Schmitt-Trigger, programmable output max-current, programmable Slew-Rate (Fast/slow), programmable internal Clamp-Diode
3. Possibility to remap the CPU ports over HAT and EXHAT
4. Integrate custom controller inside the FPGA by RTL programming (up to 16k LE)
5. Allow CPU-FPGA fast data communication by SDIO (project based, reserved resource, wont be supported on MP)
6. Drive all 4x LED integrated on UP2
7. Control the two HDMI-CEC ports on Combo HDMI connector
8. Support a hi-speed LVDS bus (one byte wide) on EXHAT (including PLL_in and PLL_out)
9. Allow CPU. to access FPGA.JTAG for fw upload
10. Allow CPU to control/monitor FPGA service signals (nSTATUS, nCONFIG, CONF_DONE, CONFIG_SEL, FPGA_RST, FPGA_OE)
11. Trigger CPU-wake event on CPU (BT_HOST_WAKE)
12. Employ 2x CPU.OSC port to get external clock source

The UP2-release will include an FPGA-fw integrating “HAT-Controller v1.0” including following functionalities:

1. CPU-FPGA bit-bang comm port used to program CONF-reg of HAT-ctrl (same as UP1)
2. Support of HAT-module (2nd SPI port will be supported for APL-I model only).
3. Support of internal Pull-UP/DWN, Schmidt Trigger, Clamp-Diode only in static mode (cannot be changed during run-time but only by full FPGA-fw recompile)
4. HDMI-CEC controller
5. CPU-fw upload
6. Control GPIO on EXHAT
7. Control 4xLED

What will not be supported on HAT-Controller v1.0 release:

1. SPI slave-port to configure the HAT-controller

Such functionalities will be activated on project-base and will require extra-NRE

2.1 HAT-controller: CONF -register

Below the definition of CONF-register of the HAT-controller integrated in “HAT-controller fw v1.0” release:

CONF_REG(52) to CONF_REG(71) refer to table below:

BIT#	*CONF-reg bit-name*	*Default value*	*description*
48-51	RSVD		
52-55	FW-ID	“0000”	**FW-ID code (READ-ONLY REGISTER)**
56	LED1 blue	0	**“1”** = LED lit ON ; “0” LED lit OFF; (signal on schema “LED0_3V3”)
57	LED2 yellow	0	**“1”** = LED lit ON ; “0” LED lit OFF; (signal on schema “LED1_3V3”)
58	LED3 green	0	**“1”** = LED lit ON ; “0” LED lit OFF; (signal on schema “LED2_3V3”)
59	LED4 red	0	**“1”** = LED lit ON ; “0” LED lit OFF; (signal on schema “LED3_3V3”)
60	Enable-I2C0	0	**“1” = enable I2C0 ; “0” = disable (default)**
61	Enable-I2C1	0	**“1” = enable I2C1 ; “0” = disable (default)**
62 RSVD	*Enable-CEC0*	*0*	***“1” = enable CEC0 ; “0” = disable (default)***
63 RSVD	*HAT-mode1*	*0*	***“1” = enable HAT-mode1; “0” = enable HAT-mode(default)***
64 RSVD	*HAT-mode2*	*0*	***“1” = enable HAT-mode2; “0” = enable HAT-mode(default)***
65	… spare …		
66	… spare …		
67	… spare …		
68	… spare …		
69	… spare …		
70	… spare …		
71	… spare …		

The FPGA buffer direction (CPU to HAT /or/ HAT to CPU) follows this rule:
value “0” (default) = CPU to HAT; value “1”= HAT to CPU

>>>> note; the schema doesnt employ HAT conventional signal naming; but in this document WE MUST REFER TO SCHEMA NAMING CONVENTIONs ONLY

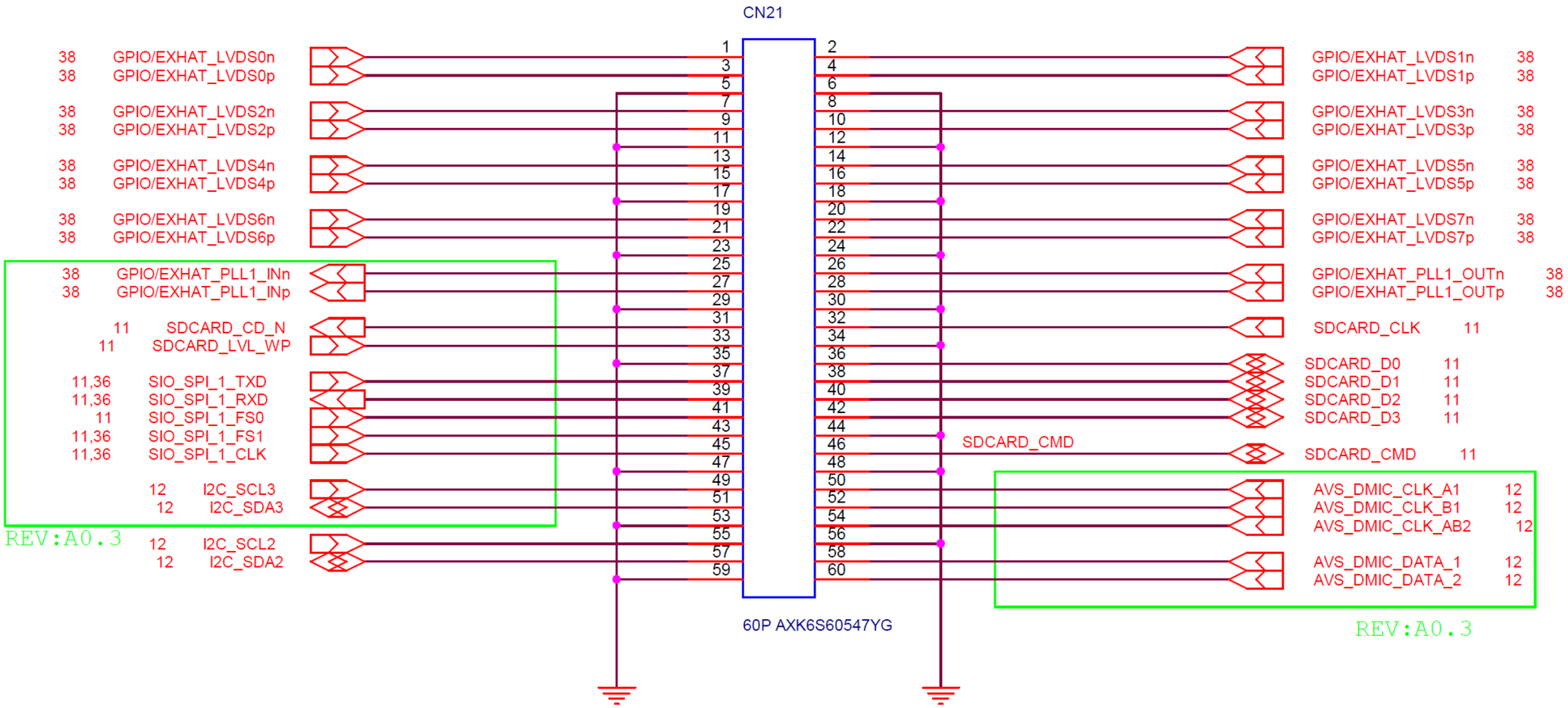
CPU/FPGA-port	CPU-signal (same name of schematic)	HAT-ctrl **CONF_REG()**	RTL signal name CPU-side	**HAT-(default)> SPI2 = GPIO**	*HAT-(mode1)> SPI2 active*	*HAT-(mode2)*	mapped on RTL signal name HAT/EXHAT side **(HAT-default)**
UART1	GPIO_43:LPSS_UART1_TXD	0	bidir_cpu0	HAT_UART1_TXD	*dont change*	*dont change*	**bidir_HAT0**
	GPIO_42:LPSS_UART1_RXD	1	bidir_cpu1	HAT_UART1_RXD	*dont change*	*dont change*	**bidir_HAT1**
	GPIO_44:LPSS_UART1_RTS	2	bidir_cpu2	HAT_UART1_R_RTS	*HAT_GPIO4*	*dont change*	**bidir_HAT2**
	GPIO_45:LPSS_UART1_CTS	3	bidir_cpu3	HAT_UART1_CTS	*HAT_R_GPIO3*	*dont change*	**bidir_HAT3**
SPI2 (on APL-I only)	GPIO_123:SIO_SPI_2_TXD	4	bidir_cpu4	HAT_R_GPIO1	*HAT_I2S6_SDI*	*dont change*	**bidir_HAT4**
	GPIO_122:SIO_SPI_2_RXD	5	bidir_cpu5	HAT_R_GPIO2	*HAT_I2S6_WS_SYNC*	*dont change*	**bidir_HAT5**
	GPIO_121:SIO_SPI_2_FS2	6	bidir_cpu6	HAT_R_GPIO3	*HAT_UART1_CTS*	*dont change*	**bidir_HAT6**
	GPIO_120:SIO_SPI_2_FS1	7	bidir_cpu7	HAT_GPIO4	*HAT_UART1_R_RTS*	*dont change*	**bidir_HAT7**
	GPIO_119:SIO_SPI_2_FS0	8	bidir_cpu8		*HAT_I2S6_BCLK*	*dont change*	bidir_EXHAT18
	GPIO_118:SIO_SPI_2_CLK	9	bidir_cpu9		*HAT_I2S6_SDO*	*dont change*	bidir_EXHAT19
I2S2 (new on EVT3)	AVS_I2S2_BCLK	10	bidir_cpu10	*HAT_SPI_1_TXD(* HAT_GPIO9*)*	*dont change*	*dont change*	**bidir_HAT8**
	AVS_I2S2_SDO	11	bidir_cpu11	*HAT_SPI_1_RXD(* HAT_GPIO8*)*	*dont change*	*dont change*	**bidir_HAT9**
	PWM3	12	bidir_cpu12	*HAT_SPI_1_FS1(* HAT_GPIO7*)*	*dont change*	*dont change*	**bidir_HAT10**
	AVS_I2S2_WS_SYNC	13	bidir_cpu13	*HAT_SPI_1_FS0(* HAT_GPIO6*)*	*dont change*	*dont change*	**bidir_HAT11**
	AVS_I2S2_SDI	14	bidir_cpu14	*HAT_SPI_1_CLK(* HAT_GPIO5*)*	*dont change*	*dont change*	**bidir_HAT12**
SPI0	GPIO_110:SIO_SPI_0_TXD	15	bidir_cpu15	HAT_SPI_0_TXD	*dont change*	*dont change*	**bidir_HAT13**
	GPIO_109:SIO_SPI_0_RXD	16	bidir_cpu16	HAT_SPI_0_RXD	*dont change*	*dont change*	**bidir_HAT14**
	GPIO_106:SIO_SPI_0_FS1	17	bidir_cpu17	HAT_SPI_0_FS1	*dont change*	*dont change*	**bidir_HAT15**
	GPIO_105:SIO_SPI_0_FS0	18	bidir_cpu18	HAT_SPI_0_FS0	*dont change*	*dont change*	**bidir_HAT16**
	GPIO_104:SIO_SPI_0_CLK	19	bidir_cpu19	HAT_SPI_0_CLK	*dont change*	*dont change*	**bidir_HAT17**
I2C0	GPIO_125:LPSS_I2C0_SCL	20	bidir_cpu20	HAT_I2C0_SCL	*dont change*	*dont change*	**bidir_HAT18**
	GPIO_124:LPSS_I2C0_SDA	21	bidir_cpu21	HAT_I2C0_SDA	*dont change*	*dont change*	**bidir_HAT19**
I2C1	GPIO_127:LPSS_I2C1_SCL	22	bidir_cpu22	HAT_I2C1_SCL	*dont change*	*dont change*	**bidir_HAT20**
	GPIO_126:LPSS_I2C1_SDA	23	bidir_cpu23	HAT_I2C1_SDA	*dont change*	*dont change*	**bidir_HAT21**
PWM1	GPIO_35:PWM1	24	bidir_cpu24	HAT_PWM1	*dont change*	*dont change*	**bidir_HAT22**
PMW0	GPIO_34:PWM0	25	bidir_cpu25	HAT_PWM0	*dont change*	*HAT_I2S6_BCLK*	**bidir_HAT23**
GPIO / ISH-GPIO	GPIO_33:**ISH_GPIO_15 (shared with PMIC_IRQ_N)**	26	bidir_cpu26	GPIO/EXHAT_LVDS0n	*dont change*	*dont change*	bidir_EXHAT0
	GPIO_32:**ISH_GPIO_14**	27	bidir_cpu27	GPIO/EXHAT_LVDS0p	*dont change*	*dont change*	bidir_EXHAT1
	GPIO_31:**ISH_GPIO_13**	28	bidir_cpu28	GPIO/EXHAT_LVDS1n	*dont change*	*dont change*	bidir_EXHAT2
	GPIO_30:**ISH_GPIO_12**	29	bidir_cpu29	GPIO/EXHAT_LVDS1p	*dont change*	*dont change*	bidir_EXHAT3
	GPIO_29:**ISH_GPIO_11**	30	bidir_cpu30	GPIO/EXHAT_LVDS2n	*dont change*	*dont change*	bidir_EXHAT4
	GPIO_28:**ISH_GPIO_10**	31	bidir_cpu31	GPIO/EXHAT_LVDS2p	*dont change*	*dont change*	bidir_EXHAT5
	GPIO_155:**ISH_GPIO_9**/SPKR	32	bidir_cpu32	GPIO/EXHAT_LVDS3n	*dont change*	*dont change*	bidir_EXHAT6
	GPIO_154:**ISH_GPIO_8**/MEMHOT_N	33	bidir_cpu33	GPIO/EXHAT_LVDS3p	*dont change*	*dont change*	bidir_EXHAT7
	GPIO_153:**ISH_GPIO_7**/AVS_I2S5_SDO/**LPSS_UART2_CTS_N**	34	bidir_cpu34	GPIO/EXHAT_LVDS4n	*dont change*	*dont change*	bidir_EXHAT8
	GPIO_152:**ISH_GPIO_6**/AVS_I2S5_SDI/**LPSS_UART2_RTS_N**	35	bidir_cpu35	GPIO/EXHAT_LVDS4p	*dont change*	*dont change*	bidir_EXHAT9
	GPIO_151:**ISH_GPIO_5**/AVS_I2S5_WS_SYNC/**LPSS_UART2_TXD**	36	bidir_cpu36	GPIO/EXHAT_LVDS5n	*dont change*	*dont change*	bidir_EXHAT10
	GPIO_150:**ISH_GPIO_4**/AVS_I2S5_BCLK/**LPSS_UART2_RXD**	37	bidir_cpu37	GPIO/EXHAT_LVDS5p	*dont change*	*dont change*	bidir_EXHAT11
I2S6 / ISH-GPIO / HDA	GPIO_149:**ISH_GPIO_3**/AVS_I2S6_SDO/**AVS_HDA_SDO**	38	bidir_cpu38	HAT_I2S6_SDO	*dont change*	*dont change*	**bidir_HAT24**
	GPIO_148:**ISH_GPIO_2**/AVS_I2S6_SDI/**AVS_HDA_SDI**	39	bidir_cpu39	HAT_I2S6_SDI	*HAT_R_GPIO1*	*dont change*	**bidir_HAT25**
	GPIO_147:**ISH_GPIO_1**/AVS_I2S6_WS_SYNC/**AVS_HDA_WS_SYNC**	40	bidir_cpu40	HAT_I2S6_WS_SYNC	*HAT_R_GPIO2*	*dont change*	**bidir_HAT26**
	GPIO_146:**ISH_GPIO_0**/AVS_I2S6_BCLK/**AVS_HDA_BCLK**	41	bidir_cpu41	HAT_I2S6_BCLK	*dont change*	*HAT_PWM0*	**bidir_HAT27**
ISH-I2C0	GPIO_134:LPSS_I2C5_SDA/ISH_I2C0_SDA	42	bidir_cpu42	GPIO/EXHAT_LVDS6n	*dont change*	*dont change*	bidir_EXHAT12
	GPIO_135:LPSS_I2C5_SCL/ISH_I2C0_SCL	43	bidir_cpu43	GPIO/EXHAT_LVDS6p	*dont change*	*dont change*	bidir_EXHAT13
ISH-I2C1	GPIO_136:LPSS_I2C6_SDA/ISH_I2C1_SDA	44	bidir_cpu44	GPIO/EXHAT_LVDS7n	*dont change*	*dont change*	bidir_EXHAT14
	GPIO_137:LPSS_I2C6_SCL/ISH_I2C1_SCL	45	bidir_cpu45	GPIO/EXHAT_LVDS7p	*dont change*	*dont change*	bidir_EXHAT15
SDIO (on APL-I only)	GPIO_166:SDIO_CLK	46	bidir_cpu46		*dont change*	*dont change*	bidir_EXHAT16
	GPIO_167:SDIO_D0		spare0				**FPGA-comm(SPARE SIGNAL0)**
	GPIO_168:SDIO_D1		data_out				**FPGA-comm(data_out)**
	GPIO_169:SDIO_D2		data_in				**FPGA-comm(data_in)**
	GPIO_170:SDIO_D3		stb				**FPGA-comm(strobe)**
	GPIO_171:SDIO_CMD		clr				**FPGA-comm(start)**
	GPIO_183:SDIO_PWR_DOWN_N		spare1				**FPGA-comm(SPARE SIGNAL1)**

FPGA-JTAG	JTAG-TCK (connected to CPU-GPIO-HDMI-CEC)		hmdi-cec_port				JTAG/HDMI-CEC port
	JTAG-TDO (connected to CPU-GPIO-HDMI-CEC)		hmdi-cec_port				
	JTAG-TMS (connected to CPU-GPIO-HDMI-CEC)		hmdi-cec_port				
	JTAG-TDI (connected to CPU-GPIO-HDMI-CEC)		hmdi-cec_port				
MISC	GPIO_84:**AVS_I2S2_MCLK**/AVS_HDA_RST_N	47	bidir_cpu47				bidir_EXHAT17
	GPIO_217:FPGA(FPGA_RST)		FPGA_rst				**FPGA-comm(rst)**
	GPIO_216:FPGA(FPGA_OE)		FPGA_oe				**FPGA-comm(oe)**
	OSC_CLK_OUT_0		OSC_CLK_OUT_0				; clock_in0
	OSC_CLK_OUT_1		OSC_CLK_OUT_1				; clock_in1
	GPIO_218:FPGA(nSTATUS)		nSTATUS				; all these PINS could be enabled as FPGA control pins or disabled to be used as user-IO
	GPIO_27:FPGA(nCONFIG) > FPGA_fw_reload		nCONFIG				
	GPIO_17:FPGA(CONF_DONE)		CONF_DONE				
	GPIO_19:FPGA(CONFIG_SEL)		CONFIG_SEL				; could be enabled or user-IO
	GPIO_18:FPGA(BT_HOST_WAKE)		BT_HOST_WAKE				; hi-Z
	GPIO_10 (CPU direct_WAKE:(BT_HOST_WAKE))						
LPC	LPC_CLKOUT0		LPC_bus				
	LPC_FRAME		LPC_bus				
	LPC_AD3		LPC_bus				
	LPC_AD2		LPC_bus				
	LPC_AD1		LPC_bus				
	LPC_AD0		LPC_bus				
	LPC_CLKRUN		LPC_bus				
	LPC_SERIRQ		LPC_bus				
		48-55	RSVD				
		56	LED1 blue				
		57	LED2 yellow				
		58	LED3 green				
		59	LED4 red				
		60	I2C_VLS_OE0				
		61	I2C_VLS_OE1				
		62	Enable-CEC0				
		63	*HAT-mode1*				
		64	*HAT-mode2*				

View of HAT-connector

	EMUTEX request (signal names from Rpi)	UP2 board (**signal names from schema**)	UP-board			UP-board	UP2 board(**signal names from schema**)	EMUTEX request (signal names from Rpi)	
		3V3	3V3	1	2	5V	5V		
bidir_HAT21	I2C1_SDA	HAT_I2C1_SDA	GPIO1/I2C1_SDA	3	4	5V	5V		
bidir_HAT20	I2C1_SCL	HAT_I2C1_SCL	GPIO2/I2C1_SCL	5	6	Ground	Ground		
bidir_HAT4	GPIO1	HAT_ANALOG_IN1-HAT_R_GPIO1	GPIO3	7	8	GPIO16/UART_TX	HAT_UART1_TXD	UART1_TXD	**bidir_HAT0**
		Ground	Ground	9	10	GPIO17/UART_RX	HAT_UART1_RXD	UART1_RXD	**bidir_HAT1**
bidir_HAT2	UART1_RTS | SPI_1_FS1*	HAT_ANALOG_IN2-HAT_UART1_R_RTS	GPIO4	11	12	GPIO18/I2S_CLK	HAT_I2S6_BCLK	I2S6_BCLK | SPI_1_FS0* | PWM0*	**bidir_HAT27**
bidir_HAT5	GPIO2	HAT_ANALOG_IN3-HAT_R_GPIO2	GPIO5	13	14	Ground	Ground		
bidir_HAT6	GPIO3	HAT_ANALOG_IN4-HAT_R_GPIO3	GPIO6	15	16	GPIO19	HAT_SPI_1_FS1	GPIO7	**bidir_HAT10**
		3V3	3V3	17	18	GPIO20	HAT_SPI_1_RXD	GPIO8	**bidir_HAT9**
bidir_HAT13	SPI_0_TXD	HAT_SPI_0_TXD	GPIO7/SPI_MOSI	19	20	Ground	Ground		
bidir_HAT14	SPI_0_RXD	HAT_SPI_0_RXD	GPIO8/SPI_MISO	21	22	GPIO21	HAT_SPI_1_TXD	GPIO9	**bidir_HAT8**
bidir_HAT17	SPI_0_CLK	HAT_SPI_0_CLK	GPIO9/SPI_CLK	23	24	GPIO22/SPI_CS0N	HAT_SPI_0_FS0	SPI_0_FS0	**bidir_HAT16**
		Ground	Ground	25	26	GPIO23/SPI_CS1N	HAT_SPI_0_FS1	SPI_0_FS1	**bidir_HAT15**
bidir_HAT19	I2C0_SDA	HAT_I2C0_SDA	GPIO10/I2C0_SDA	27	28	GPIO24/I2C0_SCL	HAT_I2C0_SCL	I2C0_SCL	**bidir_HAT18**
bidir_HAT7	GPIO4	HAT_GPIO4	GPIO11	29	30	Ground	Ground		
bidir_HAT12	GPIO5	HAT_SPI_1_CLK	GPIO12	31	32	GPIO25/PWM0	HAT_PWM0	PWM0	**bidir_HAT23**
bidir_HAT22	PMW1	HAT_PWM1	GPIO13/PWM1	33	34	Ground	Ground		
bidir_HAT26	I2S6_WS_SYNC | SPI_1_RXD*	HAT_I2S6_WS_SYNC	GPIO14/I2S_FRM	35	36	GPIO26	HAT_UART1_CTS	UART1_CTS | SPI_1_FS2*	**bidir_HAT3**
bidir_HAT11	GPIO6	HAT_SPI_1_FS0	GPIO15	37	38	GPIO27/I2S_DATAIN	HAT_I2S6_SDI	I2S6_SDI | SPI_1_TXD*	**bidir_HAT25**
		Ground	Ground	39	40	GPIO28/I2S_DATAOUT	HAT_I2S6_SDO	I2S6_SDO | SPI_1_CLK*	**bidir_HAT24**

View of EXHAT-connector



2.2 HAT-controller programming

The CPU can access the CONF-reg inside the HAT controller to perform a “write-readback sequence” allowing to both write a new value inside the CONF-reg and read it back before FPGA pin assumes the new value. The comm interface between CPU and FPGA will use a “bit-bang” protocol using following CPU-GPIOs:

FPGA-RST (GPIO_217.PIN- M29): used to fully reset all register of HAT-controller; signal has pull-dwn

FPGA-OE(GPIO_216.PIN- P30): when asserted (0), all FPGA-PIN will be set to High-Z (but Dout); when de-asserted(1) the FPGA-PIN buffers will assume the direction (In/or/ Out) determined by the CONF-reg; signal has pull-dwn

Clear/Start (GPIO_171.PIN- T57): used to clear all registers but the CONF_REG() of HAT-controller register so allowing to start write-read-back sequence without affecting the internal CONF-register

Strobe(GPIO_170.PIN- T55): used as clock signal

Din(GPIO_169.PIN- T54): input data to FPGA

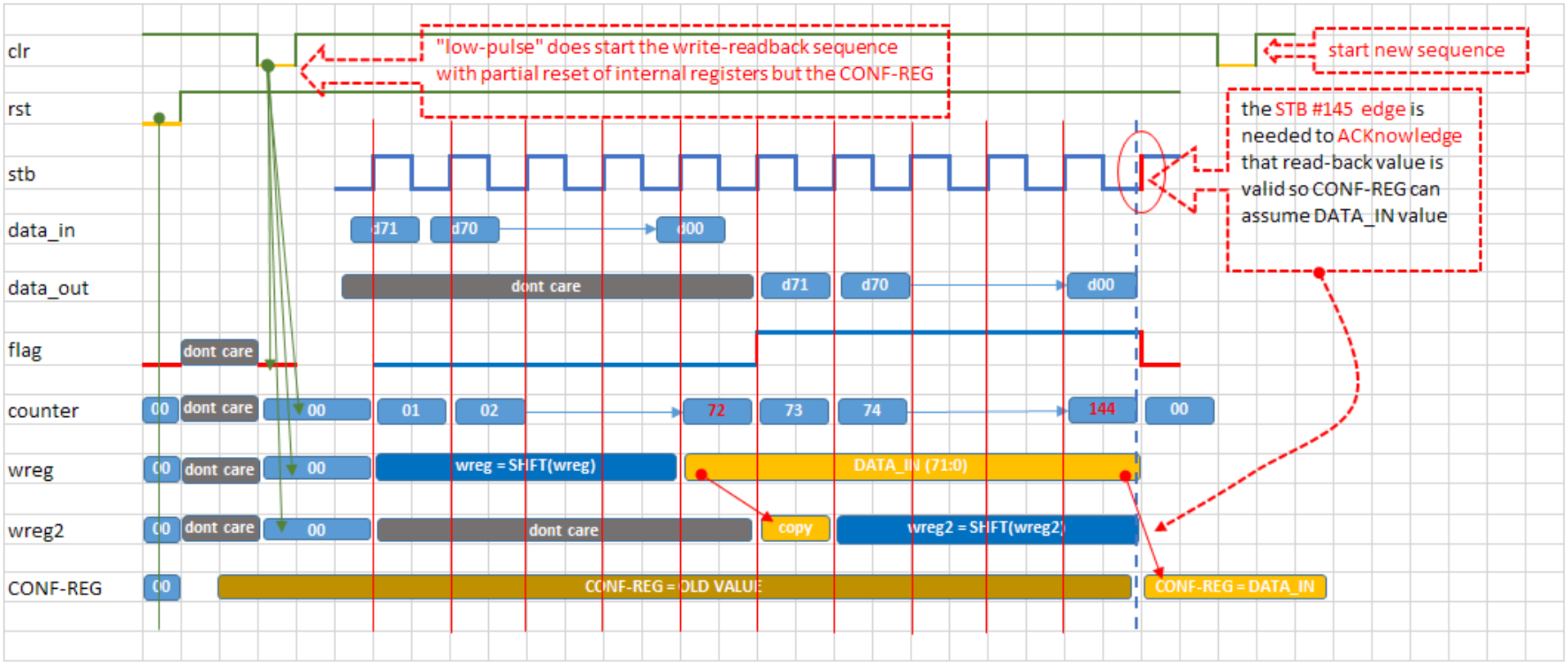
Dout(GPIO_168.PIN- P57): used for read-back to CPU the new value before latching to CONF_REG()

@BOOT: direction of all PINs is set to CPU→HAT by default; both FPGA-RST and FPGA-OE must be kept asserted (ZERO).
@FPGA-programming:

Step1. keep FPGA-OE asserted (low), release FPGA_RST (high)
Step2. program CONF_REG() using FPGA_comm bus (clear-stb-Din-Dout)
Step3. Configure CPU-GPIO according to FPGA-CPU interface configuration set on CONF_REG()
Step4. Release FPGA-OE (high)

Note: If this procedure is not performed correctly, short circuits between CPU pins and FPGA pins might occurs which would damage the CPU and/or FPGA

CONF_REG() programming by FPGA_comm port



CLR (low-pulse, asynchr) has to be generated at the beginning of any new write-readback sequence; it does perform a partial reset of HAT-controller registers (flag, wreg1, wreg2 and counter) but leaving unchanged CONF-reg to avoid affecting the HAT communication undergoing;

LAST STROBE (STB#145) is required by the HAT-controller in order to update the CONF-reg with new value

2.3 FPGA-fw upload from CPU

The FPGA will be programmed by CPU only on EVT3 version (ask driver to EMUTEX)

2.4 HAT-ctrl RTL source-code i/f

Below the RTL code “170116 up2max v0.2.0” for HAT-CONTROLLER:

```
-- ************************************************
 -- ************************************************
 -- Project name: UP2board EVT2 FGPA_fw
 -- Auth: LuigiGrasso@aaeon.eu
 -- Dept: AEU
 -- Date: 2017-01-16
 -- directory: prj_UP2\MAX 10\FPGA FW\MAX10-2K_prj2
 -- EVT2 version. test-fw according to "170116 UP2 MAX10 ref mapping EVT v1.6.xlsx"
 -- ref schema: up-apl01_a02_20161102a.pdf
 -- ************************************************
 -- ************************************************

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

ENTITY up2max IS
    PORT(

       bidir_cpu                        : INOUT STD_LOGIC_VECTOR (48 DOWNTO 0);
      bidir_hat                         : INOUT STD_LOGIC_VECTOR (27 DOWNTO 0); --
HAT_conn 28 pins
      bidir_exhat                          : INOUT STD_LOGIC_VECTOR (19 DOWNTO 0); -
- LVDS_bus(0-7) +PLL_in(0-1) +PLL_out(0-1)
      lpc_bus                           : OUT STD_LOGIC_VECTOR (7 DOWNTO 0);     --
LPC bus to be set in hi-Z
      data_in, stb, clr, spare0, spare1 : IN STD_LOGIC; -- mapped on SDIO-port
      data_out                          : OUT STD_LOGIC;     -- mapped on SDIO-port
      oe_fpga, rst_fpga                    : IN STD_LOGIC; -- mapped on FPGA_OE and
FPGA_RST
      OSC_CLK_OUT_0, OSC_CLK_OUT_1      : IN STD_LOGIC;
      vls_ce0, vls_ce1                  : OUT STD_LOGIC;     -- mapped on
GPIO_EXHAT_LVDS8n/p and enabling VLS
```

```
        GPIO_TDI, GPIO_TMS, GPIO_TCK        : IN STD_LOGIC;     --
HMDI/cec port (MUX with JTAG) *** when JTAG sharing enabled >> this has to be IN-
port***  to avoid Quartus bug
        GPIO_TDO                            : OUT STD_LOGIC;    -- HDMI/cec port (MUX
with JTAG) *** when JTAG sharing enabled >> this has to be OUT-port*** to avoid
Quartus bug
 --nSTATUS, nCONFIG, CONF_DONE, : IN STD_LOGIC;     -- Dual Purpose PINS (FPGA
control pins); when disabled they can be used as user-IO
 --CONFIG_SEL                       : IN STD_LOGIC;     -- Dual Purpose PINS (FPGA
control pins); when disabled they can be used as user-IO
 --BT_HOST_WAKE                     : IN STD_LOGIC;     -- trigger CPU-WAKE EVENTs
       led_3V3                              : OUT STD_LOGIC_VECTOR (3 DOWNTO 0));


END up2max;

ARCHITECTURE up2max_a OF up2max IS

SIGNAL  count_reg               : STD_LOGIC_VECTOR (7 DOWNTO 0);  -- used to trigger
conf_reg registration
SIGNAL  conf_reg, wreg, wreg2   : STD_LOGIC_VECTOR (71 DOWNTO 0);
SIGNAL  flag1                   : STD_LOGIC; -- flag if-clauses met

BEGIN

bidir_exhat <= (others => 'Z'); -- test version: disable exhat

lpc_bus(7)      <= 'Z';
lpc_bus(6)      <= 'Z';
lpc_bus(5)      <= 'Z';
lpc_bus(4)      <= 'Z';
lpc_bus(3)      <= 'Z';
lpc_bus(2)      <= 'Z';
lpc_bus(1)      <= 'Z';
lpc_bus(0)      <= 'Z';

led_3V3(0)  <= conf_reg(56);
led_3V3(1)  <= conf_reg(57);
led_3V3(2)  <= conf_reg(58);
```

```
led_3V3(3)  <= conf_reg(59);

vls_ce0 <= conf_reg(60); -- check assert level
vls_ce1 <= conf_reg(61); -- check assert level

GPIO_TDO    <= conf_reg(62);


data_out <= wreg2(71);

-- >>>>>>>>> PROC1: reg-bank, ALU, counter

PROCESS (stb, rst_fpga, clr)
BEGIN

    IF (rst_fpga = '0') THEN
        count_reg   <= (others => '0');
        wreg        <= (others => '0');
        wreg2       <= (others => '0');
        flag1       <= '0';

        conf_reg    <= (others => '0');

    ELSIF (clr = '0') THEN
        count_reg   <= (others => '0');
        wreg        <= (others => '0');
        wreg2       <= (others => '0');
        flag1       <= '0';

    ELSIF (stb = '1' AND stb'EVENT) THEN    -- STROBE signal to latch the data_in

        IF (count_reg = "10010000") THEN -- count_reg = '144'
            count_reg <= (others => '0');
            conf_reg(71 DOWNTO 0) <= wreg(71 DOWNTO 0);
            flag1 <= '0';

        ELSIF (count_reg = "01001000") THEN     -- count_reg = '72'
            count_reg <= count_reg + "00000001";
            wreg2  <= wreg;
```

```
            flag1 <= '1';

        ELSE
            count_reg <= count_reg + "00000001";
            IF (flag1 = '0') THEN        -- write-shift
                wreg(0) <= data_in;
            wreg(71 DOWNTO 1) <= wreg(70 DOWNTO 0);
            ELSE                         -- readback-shift
                wreg2(71 DOWNTO 1) <= wreg2(70 DOWNTO 0);
            END IF;

        END IF;

    END IF;
    END PROCESS;
      -- >>>>>>>>> END_PROC1

-- >>>>>>>>> PROC2: comb logic and 3-states

PROCESS (oe_fpga, bidir_cpu, bidir_hat, bidir_exhat, conf_reg)
BEGIN

    IF (oe_fpga = '0') THEN
        bidir_hat   <= (others => 'Z');
        bidir_exhat <= (others => 'Z');
        bidir_cpu   <= (others => 'Z');

    ELSE

    bidir_cpu(48)   <= 'Z';

-- >>>>>>>>>>>                          >>>>>>>>>>>
-- >>>>>>>>>>>        28x pins HAT      >>>>>>>>>>>
-- >>>>>>>>>>>                          >>>>>>>>>>>

-- >>>>>>>>>>>
        IF( conf_reg(0) = '1') THEN
            bidir_hat(0) <= 'Z';
            bidir_cpu(0) <= bidir_hat(0);
```

```
        ELSE
            bidir_cpu(0) <= 'Z';
            bidir_hat(0) <= bidir_cpu(0);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(1) = '1') THEN
            bidir_hat(1) <= 'Z';
            bidir_cpu(1) <= bidir_hat(1);
        ELSE
            bidir_cpu(1) <= 'Z';
            bidir_hat(1) <= bidir_cpu(1);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(2) = '1') THEN
            bidir_hat(2) <= 'Z';
            bidir_cpu(2) <= bidir_hat(2);
        ELSE
            bidir_cpu(2) <= 'Z';
            bidir_hat(2) <= bidir_cpu(2);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(3) = '1') THEN
            bidir_hat(3) <= 'Z';
            bidir_cpu(3) <= bidir_hat(3);
        ELSE
            bidir_cpu(3) <= 'Z';
            bidir_hat(3) <= bidir_cpu(3);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(4) = '1') THEN
            bidir_hat(4) <= 'Z';
            bidir_cpu(4) <= bidir_hat(4);
        ELSE
            bidir_cpu(4) <= 'Z';
            bidir_hat(4) <= bidir_cpu(4);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(5) = '1') THEN
            bidir_hat(5) <= 'Z';
```

```
            bidir_cpu(5) <= bidir_hat(5);
        ELSE
            bidir_cpu(5) <= 'Z';
            bidir_hat(5) <= bidir_cpu(5);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(6) = '1') THEN
            bidir_hat(6) <= 'Z';
            bidir_cpu(6) <= bidir_hat(6);
        ELSE
            bidir_cpu(6) <= 'Z';
            bidir_hat(6) <= bidir_cpu(6);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(7) = '1') THEN
            bidir_hat(7) <= 'Z';
            bidir_cpu(7) <= bidir_hat(7);
        ELSE
            bidir_cpu(7) <= 'Z';
            bidir_hat(7) <= bidir_cpu(7);
        END IF;
-- >>>>>>>>>>>
-- PIN_8 / PIN_9
-- >>>>>>>>>>>
        IF( conf_reg(10) = '1') THEN
            bidir_hat(8) <= 'Z';
            bidir_cpu(10) <= bidir_hat(8);
        ELSE
            bidir_cpu(10) <= 'Z';
            bidir_hat(8) <= bidir_cpu(10);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(11) = '1') THEN
            bidir_hat(9) <= 'Z';
            bidir_cpu(11) <= bidir_hat(9);
        ELSE
            bidir_cpu(11) <= 'Z';
            bidir_hat(9) <= bidir_cpu(11);
        END IF;
```

```
-- >>>>>>>>>>>
        IF( conf_reg(12) = '1') THEN
            bidir_hat(10) <= 'Z';
            bidir_cpu(12) <= bidir_hat(10);
        ELSE
            bidir_cpu(12) <= 'Z';
            bidir_hat(10) <= bidir_cpu(12);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(13) = '1') THEN
            bidir_hat(11) <= 'Z';
            bidir_cpu(13) <= bidir_hat(11);
        ELSE
            bidir_cpu(13) <= 'Z';
            bidir_hat(11) <= bidir_cpu(13);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(14) = '1') THEN
            bidir_hat(12) <= 'Z';
            bidir_cpu(14) <= bidir_hat(12);
        ELSE
            bidir_cpu(14) <= 'Z';
            bidir_hat(12) <= bidir_cpu(14);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(15) = '1') THEN
            bidir_hat(13) <= 'Z';
            bidir_cpu(15) <= bidir_hat(13);
        ELSE
            bidir_cpu(15) <= 'Z';
            bidir_hat(13) <= bidir_cpu(15);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(16) = '1') THEN
            bidir_hat(14) <= 'Z';
            bidir_cpu(16) <= bidir_hat(14);
        ELSE
            bidir_cpu(16) <= 'Z';
            bidir_hat(14) <= bidir_cpu(16);
```

```
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(17) = '1') THEN
            bidir_hat(15) <= 'Z';
            bidir_cpu(17) <= bidir_hat(15);
        ELSE
            bidir_cpu(17) <= 'Z';
            bidir_hat(15) <= bidir_cpu(17);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(18) = '1') THEN
            bidir_hat(16) <= 'Z';
            bidir_cpu(18) <= bidir_hat(16);
        ELSE
            bidir_cpu(18) <= 'Z';
            bidir_hat(16) <= bidir_cpu(18);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(19) = '1') THEN
            bidir_hat(17) <= 'Z';
            bidir_cpu(19) <= bidir_hat(17);
        ELSE
            bidir_cpu(19) <= 'Z';
            bidir_hat(17) <= bidir_cpu(19);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(60)='1') THEN  -- enable I2C0 / SCL
            bidir_hat(18) <= 'Z';
            bidir_cpu(20) <= 'Z';
        ELSIF( conf_reg(20) = '1') THEN
            bidir_hat(18) <= 'Z';
            bidir_cpu(20) <= bidir_hat(18);
        ELSE
            bidir_cpu(20) <= 'Z';
            bidir_hat(18) <= bidir_cpu(20);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(60)='1') THEN  -- enable I2C0 / SDA
            bidir_hat(19) <= 'Z';
```

```
            bidir_cpu(21) <= 'Z';
        ELSIF( conf_reg(21) = '1') THEN
            bidir_hat(19) <= 'Z';
            bidir_cpu(21) <= bidir_hat(19);
        ELSE
            bidir_cpu(21) <= 'Z';
            bidir_hat(19) <= bidir_cpu(21);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(61)='1') THEN -- enable I2C1 / SCL
            bidir_hat(20) <= 'Z';
            bidir_cpu(22) <= 'Z';
        ELSIF( conf_reg(22) = '1') THEN
            bidir_hat(20) <= 'Z';
            bidir_cpu(22) <= bidir_hat(20);
        ELSE
            bidir_cpu(22) <= 'Z';
            bidir_hat(20) <= bidir_cpu(22);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(61)='1') THEN -- enable I2C1 / SDA
            bidir_hat(21) <= 'Z';
            bidir_cpu(23) <= 'Z';
        ELSIF( conf_reg(23) = '1') THEN
            bidir_hat(21) <= 'Z';
            bidir_cpu(23) <= bidir_hat(21);
        ELSE
            bidir_cpu(23) <= 'Z';
            bidir_hat(21) <= bidir_cpu(23);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(24) = '1') THEN
            bidir_hat(22) <= 'Z';
            bidir_cpu(24) <= bidir_hat(22);
        ELSE
            bidir_cpu(24) <= 'Z';
            bidir_hat(22) <= bidir_cpu(24);
        END IF;
-- >>>>>>>>>>>
```

```
        IF( conf_reg(25) = '1') THEN
            bidir_hat(23) <= 'Z';
            bidir_cpu(25) <= bidir_hat(23);
        ELSE
            bidir_cpu(25) <= 'Z';
            bidir_hat(23) <= bidir_cpu(25);
        END IF;
-- >>>>>>>>>>>
-- PIN_26 / PIN_37

-- >>>>>>>>>>>
        IF( conf_reg(38) = '1') THEN
            bidir_hat(24) <= 'Z';
            bidir_cpu(38) <= bidir_hat(24);
        ELSE
            bidir_cpu(38) <= 'Z';
            bidir_hat(24) <= bidir_cpu(38);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(39) = '1') THEN
            bidir_hat(25) <= 'Z';
            bidir_cpu(39) <= bidir_hat(25);
        ELSE
            bidir_cpu(39) <= 'Z';
            bidir_hat(25) <= bidir_cpu(39);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(40) = '1') THEN
            bidir_hat(26) <= 'Z';
            bidir_cpu(40) <= bidir_hat(26);
        ELSE
            bidir_cpu(40) <= 'Z';
            bidir_hat(26) <= bidir_cpu(40);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(41) = '1') THEN
            bidir_hat(27) <= 'Z';
            bidir_cpu(41) <= bidir_hat(27);
        ELSE
```

```
            bidir_cpu(41) <= 'Z';
            bidir_hat(27) <= bidir_cpu(41);
        END IF;

-- >>>>>>>>>>>                          >>>>>>>>>>>
-- >>>>>>>>>>>      20xpins EXHAT       >>>>>>>>>>>
-- >>>>>>>>>>>                          >>>>>>>>>>>

-- >>>>>>>>>>>
        IF( conf_reg(26) = '1') THEN
            bidir_exhat(0) <= 'Z';
            bidir_cpu(26) <= bidir_exhat(0);
        ELSE
            bidir_cpu(26) <= 'Z';
            bidir_exhat(0) <= bidir_cpu(26);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(27) = '1') THEN
            bidir_exhat(1) <= 'Z';
            bidir_cpu(27) <= bidir_exhat(1);
        ELSE
            bidir_cpu(27) <= 'Z';
            bidir_exhat(1) <= bidir_cpu(27);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(28) = '1') THEN
            bidir_exhat(2) <= 'Z';
            bidir_cpu(28) <= bidir_exhat(2);
        ELSE
            bidir_cpu(28) <= 'Z';
            bidir_exhat(2) <= bidir_cpu(28);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(29) = '1') THEN
            bidir_exhat(3) <= 'Z';
            bidir_cpu(29) <= bidir_exhat(3);
        ELSE
            bidir_cpu(29) <= 'Z';
            bidir_exhat(3) <= bidir_cpu(29);
```

```
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(30) = '1') THEN
            bidir_exhat(4) <= 'Z';
            bidir_cpu(30) <= bidir_exhat(4);
        ELSE
            bidir_cpu(30) <= 'Z';
            bidir_exhat(4) <= bidir_cpu(30);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(31) = '1') THEN
            bidir_exhat(5) <= 'Z';
            bidir_cpu(31) <= bidir_exhat(5);
        ELSE
            bidir_cpu(31) <= 'Z';
            bidir_exhat(5) <= bidir_cpu(31);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(32) = '1') THEN
            bidir_exhat(6) <= 'Z';
            bidir_cpu(32) <= bidir_exhat(6);
        ELSE
            bidir_cpu(32) <= 'Z';
            bidir_exhat(6) <= bidir_cpu(32);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(33) = '1') THEN
            bidir_exhat(7) <= 'Z';
            bidir_cpu(33) <= bidir_exhat(7);
        ELSE
            bidir_cpu(33) <= 'Z';
            bidir_exhat(7) <= bidir_cpu(33);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(34) = '1') THEN
            bidir_exhat(8) <= 'Z';
            bidir_cpu(34) <= bidir_exhat(8);
        ELSE
            bidir_cpu(34) <= 'Z';
```

```
            bidir_exhat(8) <= bidir_cpu(34);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(35) = '1') THEN
            bidir_exhat(9) <= 'Z';
            bidir_cpu(35) <= bidir_exhat(9);
        ELSE
            bidir_cpu(35) <= 'Z';
            bidir_exhat(9) <= bidir_cpu(35);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(36) = '1') THEN
            bidir_exhat(10) <= 'Z';
            bidir_cpu(36) <= bidir_exhat(10);
        ELSE
            bidir_cpu(36) <= 'Z';
            bidir_exhat(10) <= bidir_cpu(36);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(37) = '1') THEN
            bidir_exhat(11) <= 'Z';
            bidir_cpu(37) <= bidir_exhat(11);
        ELSE
            bidir_cpu(37) <= 'Z';
            bidir_exhat(11) <= bidir_cpu(37);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(42) = '1') THEN
            bidir_exhat(12) <= 'Z';
            bidir_cpu(42) <= bidir_exhat(12);
        ELSE
            bidir_cpu(42) <= 'Z';
            bidir_exhat(12) <= bidir_cpu(42);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(43) = '1') THEN
            bidir_exhat(13) <= 'Z';
            bidir_cpu(43) <= bidir_exhat(13);
        ELSE
```

```
            bidir_cpu(43) <= 'Z';
            bidir_exhat(13) <= bidir_cpu(43);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(44) = '1') THEN
            bidir_exhat(14) <= 'Z';
            bidir_cpu(44) <= bidir_exhat(14);
        ELSE
            bidir_cpu(44) <= 'Z';
            bidir_exhat(14) <= bidir_cpu(44);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(45) = '1') THEN
            bidir_exhat(15) <= 'Z';
            bidir_cpu(45) <= bidir_exhat(15);
        ELSE
            bidir_cpu(45) <= 'Z';
            bidir_exhat(15) <= bidir_cpu(45);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(46) = '1') THEN
            bidir_exhat(16) <= 'Z';
            bidir_cpu(46) <= bidir_exhat(16);
        ELSE
            bidir_cpu(46) <= 'Z';
            bidir_exhat(16) <= bidir_cpu(46);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(47) = '1') THEN
            bidir_exhat(17) <= 'Z';
            bidir_cpu(47) <= bidir_exhat(17);
        ELSE
            bidir_cpu(47) <= 'Z';
            bidir_exhat(17) <= bidir_cpu(47);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(8) = '1') THEN
            bidir_exhat(18) <= 'Z';
            bidir_cpu(8) <= bidir_exhat(18);
```

```
        ELSE
            bidir_cpu(8) <= 'Z';
            bidir_exhat(18) <= bidir_cpu(8);
        END IF;
-- >>>>>>>>>>>
        IF( conf_reg(9) = '1') THEN
            bidir_exhat(19) <= 'Z';
            bidir_cpu(9) <= bidir_exhat(19);
        ELSE
            bidir_cpu(9) <= 'Z';
            bidir_exhat(19) <= bidir_cpu(9);
        END IF;
-- >>>>>>>>>>>
    END IF;

    END PROCESS;
     -- >>>>>>>>> END_PRO



END up2max_a;
```