

```
Name PDP11_main;
Assembly 0001;
Revision 0.5;
PartNo U4 ATF1508AS;
Device f1508isplcc84;
Company S100Computers.com;
Designer John Monahan;
Location CA, San Ramon;
Date 29 March 2017;
```

```
property ATMEL { xor_synthesis=on };
property ATMEL { logic_doubling=on };
property ATMEL { jtag=on };
PROPERTY ATMEL { preassign keep };
PROPERTY ATMEL { TMS_pullup=on };
PROPERTY ATMEL { TDI_pullup=on };
PROPERTY ATMEL {open_collector=ABORT};
```

```
/*
* ----- BOARD CPLD IS SETUP TO RUN AS A S100 BUS >>>> SLAVE <<<<< 80386 V2 Board(V2.31) -----
* Pin assignments
* Make all data and address outputs fast slew and all chip selects slow
* Note this version allows for 16 & 8 bit RAM access.
* However it assumes all S100 bus Port I/O data is only 8 bits wide. (Can be on odd or even addresses)
*
* V0.9a      3/29/2017          ;Tested on V0.9a board
* V0.9b      4/5/2017          ;Added DV input back to PDP CPU
*/
```

```
Pin 83 = UART_CLK_IN;          /* 20MHz --> 1.84MHz */
Pin 2  = MASTER_CLK;
Pin 81 = PHI;
```

```
Pin 4  = BAUD_CLK;            /* (156.24/16)KHz = 9600 Baud (Approx) */
Pin 5  = TMax;
Pin 6  = HOLD;
Pin 8  = pHLDA;
```

```
Pin 9  = BOARD_ACTIVE;        /* LOW when the PDP is granted complete control of the S100 Bus */
Pin 10 = XFERI;               /* Active low */
Pin 11 = XFERII;
Pin 15 = ALE;
Pin 16 = PDP_WR;
```

```
Pin 17 = SHADOW_ROM;
Pin 18 = S100_PORTS_MAP;
Pin 20 = S100_IO;
Pin 21 = P26;
```

```

Pin 22 = BUFCTL;
Pin 24 = bpSYNC;
Pin 25 = READY;

Pin 27 = bpWR;
Pin 28 = bpDBIN;
Pin 29 = bsXTRQ;
Pin 30 = bMWRT;

Pin 31 = bsHLTA;
Pin 33 = bsINTA;
Pin 34 = bsWO;
Pin 35 = bsM1;

Pin 36 = bsOUT;
Pin 37 = bsINP;
Pin 39 = bsMEMR;
Pin 40 = PHANTOM;

Pin 41 = SIXTN;                /* Low if bus can handle 16 bit data */
Pin 44 = DIRECTION_L;
Pin 46 = DIRECTION_H;
Pin 48 = LA0;

Pin 49 = LED1;                /* D1 */
Pin 50 = LED2;                /* D2 */

Pin 51 = OE_A;
Pin 52 = OE_B;
Pin 54 = OE_C;
Pin 55 = OE_D;
Pin 56 = UART_ADDRESS;       /* Active Low */
Pin 57 = ROM_ADDRESS;        /* Active Low */
Pin 58 = ROM_CS;
Pin 60 = NDEF3;
Pin 61 = NDEF1;
Pin 63 = MAP;
Pin 64 = DMR;
Pin 65 = DV;
Pin 67 = UART_CS;            /* Active LOW. Don't access the S100 bus I/O ports if PDP ia addressing UART ports */
Pin 68 = ABORT;              /* Active low, Note, Open Collector output */

Pin 69 = SCTL;                /* Active low */
Pin 70 = STRB;
Pin 73 = CONT;                /* Pulse Low ends a stretch cycle */
Pin 74 = LBS0;

```

```

Pin 75 = LBS1;

Pin 76 = LAIO3;
Pin 77 = LAIO2;
Pin 79 = LAIO1;
Pin 80 = LAIO0;
Pin 84 = P22;

Pin 1 = MASTER_RESET;          /* S100 Bus reset. Active LOW */

Pin 12 = INACTIVATE_CONTROL_LINES;
Pin 45 = INACTIVATE_DATA_LINES;

/* Need to divide the 20Hz Oscillator down to obtain the correct BAUD Rate for the UART. */
/* For 9600 Baud (156.24/16 = 9765) is close enough for most Serial ports. Never a problem with the USB/Serial adaptors */

Pinnode = [CD6..0];

CD0.t = 'b'1;                  /* 10MHz */
CD1.t = CD0;                   /* 5 MHz */
CD2.t = CD0 & CD1;            /* 2.5 MHz */
CD3.t = CD0 & CD1 & CD2;      /* 1.25 Mhz */
CD4.t = CD0 & CD1 & CD2 & CD3; /* 625 KHz */
CD5.t = CD0 & CD1 & CD2 & CD3 & CD4; /* 312 KHz */
CD6.t = CD0 & CD1 & CD2 & CD3 & CD4 & CD5; /* 156 KHz */

[CD6..0].ckmux = UART_CLK_IN;

/* BAUD_CLK = CD4;             /* ~38400 Baud */
/* BAUD_CLK = CD5;             /* ~19200 Baud */
  BAUD_CLK = CD6;             /* 156.24/16 KHz = 9765 baud */

Pinnode = [reg3..0];          /* Need a few D typer Flip Flops */

reg0.d = TMAx;                /* TMAx goes low-> high */
reg0.ckmux = !PHI;
reg0.ar = !MASTER_RESET;     /* Return to high on reset */
!HOLD = reg0;                 /* Lower S100 bus HOLD line */

!XFERI = reg0 & pHLDA;

reg1.d = reg0;
reg1.ck = pHLDA;
reg1.ar = !reg0;

```

```

!XFERII = reg1;

/* !LED2      = !XFERII; */

reg2.d = reg1;
reg2.ckmux = !PHI;
reg2.ar = !reg1;

!BOARD_ACTIVE = reg2 & MASTER_RESET;          /* Take PDP-11 out of HALT Mode */

INACTIVATE_DATA_LINES = 'b'1;                /* Just to be safe, for now, pull these unimplemented signals high */
INACTIVATE_CONTROL_LINES = 'b'1;

DMR          = 'b'1;

*/
*/

/* Note: Unlike our previous CPU boards the status lines will go to bus via a 74LS244,U27
/* not a 74LS240, so the S100 bus status signals from the CPLD will NOT be invreted here.
/* Note BUFCTL is actually BUFCTL* */

Pinnode = BUS_READ;                          /* BUS CYCLE TYPE */
Pinnode = BUS_BYTE_WRITE;
Pinnode = BUS_WORD_WRITE;
Pinnode = LBS_MEM;
Pinnode = LBS_SYS;
Pinnode = LBS_IO;
Pinnode = LSB_INT;

Pinnode = ROM_ACTIVE;

!LBS_MEM    = !LBS1 & !LBS0;                  /* Need to decode LBS0 & LBS1 */
!LBS_SYS    = !LBS1 & LBS0;
!LBS_IO     = LBS1 & !LBS0;
!LSB_INT    = LBS1 & LBS0;

!BUS_READ   = ((LAI03 & LAI02 & !LAI01 & !LAI00) /* Instruction read - request */
               # (LAI03 & !LAI02 & LAI01 & LAI00) /* Read-modify-write no bus lock */
               # (LAI03 & !LAI02 & LAI01 & !LAI00) /* Read-modify-write bus lock */
               # (LAI03 & !LAI02 & !LAI01 & LAI00) /* Data read */
               # (LAI03 & !LAI02 & !LAI01 & !LAI00)); /* Instruction read - demand */

!BUS_WORD_WRITE = !LAI03 & !LAI02 & !LAI01 & LAI00; /* Bus Word WRITE*/
!BUS_BYTE_WRITE  = !LAI03 & !LAI02 & LAI01 & LAI00; /* Bus BYTE WRITE */

```

```

bpSYNC      = STRB & !XFERII;

bmWRT      = BUFCTL & !PDP_WR & (!LBS_MEM & ROM_CS) & !XFERII;
!bpWR      = BUFCTL & !PDP_WR & ((!LBS_MEM & ROM_CS) # (!LBS_IO & UART_ADDRESS)) & !XFERII;

bsMEMR     = !BUFCTL & PDP_WR & (!LBS_MEM & ROM_CS) & !XFERII & !BUS_READ;          /* Onboard ROMS local to board */
bpDBIN     = !BUFCTL & PDP_WR & ((!LBS_MEM & ROM_CS) # (!LBS_IO & UART_ADDRESS)) & !BUS_READ & !XFERII;

!OE_D      = !bpWR & !BUS_BYTE_WRITE & LA0 & !XFERII;          /* 8 Bit Write Odd address on DAL 8-15 */
!OE_A      = bpDBIN & !LBS_IO & !LA0 & !XFERII;          /* 8 Bit I/O Read Even address on DAL 0-7 */

DIRECTION_L = BUFCTL;
!OE_B      = ((!bpWR & !BUS_WORD_WRITE & !XFERII) # (bpDBIN & !LBS_MEM & !XFERII)) /* 16 Bit Rd or Wr and 8 bit RAM Rd */
            # (!bpWR & !BUS_BYTE_WRITE & !LA0 & !XFERII);      /* 8 Bit Write Even address (DAL 0-7) */

DIRECTION_H = BUFCTL;
!OE_C      = ((!bpWR & !BUS_WORD_WRITE & !XFERII) # (bpDBIN & !XFERII));      /* 16 Bit Rd or /Wr and 8 bit Rd, DAL 8-15 */

!bsXTRQ    = bpDBIN # (!bpWR & !BUS_WORD_WRITE);          /* All reads are 16 bit. For S100 Bus BYTE Write, bsXREQ must be HIGH */

bsOUT      = !bpSYNC & BUFCTL & !PDP_WR & !LBS_IO & !XFERII & (!BUS_WORD_WRITE # !BUS_BYTE_WRITE) & UART_ADDRESS;
bsINP      = !bpSYNC & !BUFCTL & PDP_WR & !LBS_IO & !XFERII & !BUS_READ & UART_ADDRESS;

bsINTA     = 'b'0;

!bsWO      = bmWRT # bsOUT;

bsM1       = ((ALE & !XFERII & (LAI03 & LAIO2 & !LAI01 & !LAI00))          /* 1100 Instruction Read (request) */
            # (ALE & !XFERII & (LAI03 & !LAI02 & !LAI01 & !LAI00)));      /* 1000 Instruction Read (demand) */

!ROM_CS    = !ROM_ADDRESS & !ROM_ACTIVE & !XFERII;          /* Onboard ROMs CS* (Pin 20 of 28C64's) */
!PHANTOM   = !ROM_CS;          /* Prevent conflict with onboard RAM/ROM with S100 bus RAM */
*/
*/          /* For testing will have RAM/ROMs at C000H (140,000 Octal) */

!UART_CS   = !UART_ADDRESS & !LBS_IO;          /* UART is 3FFF70-3FFF77 (only) (No S100 bus ports allowed) */
*/

```

```

!CONT                = !SCTL & !READY & !XFERII;                /* S100 Bus wait states required, stretch PDP-11 Cycle */
                                                                /* READY* goes high if there are wait states U34B */
                                                                /* Also required to keep CPU in hold state after power up

until it has bus */

DV                  = !SCTL & !BUFCTL;                            /* For stretched cycles to latch data into the CPU */

E000H. Before */
                                                                /* OS systems like RT11 they assume they have RAM from 0 -
                                                                /* transferring control to the OS in RAM after loading, we
                                                                /* the boards onboard ROM's at C000H-DFFFH. Inputting from
                                                                /* Be sure you are not running the code to do so from the

ROM */
reg3.d = 'b'1;
reg3.ck = !SHADOW_ROM;
reg3.ar = !MASTER_RESET # !TMax;
ROM_ACTIVE = reg3;
                                                                /* SHADOW_ROM comes from pin 18 of PDP_GAL0 */

!LED1                = !ROM_ACTIVE;

!LED2                = !CONT;

```