EE318 Electronic Design Lab Project Report, EE Dept., IIT Bombay, April 2007

High speed USB interface for digital processor systems

Group No: D11

Chaitanya Rao (04002028) < raochaitanya@ee.iitb.ac.in >
Anshul Jhawar (04002025) < anshuljha@ee.iitb.ac.in >
Atit Parikh (04d07001) < atitparikh@ee.iitb.ac.in >

Supervisor: Prof. M. C. Chandorkar

Abstract

This project attempts to provide a high speed USB (480 Mbps) interface for a TMS320VC33 digital processor system. The interface will support user program download and debugging, as well as data exchange between a host computer and the processor while user programs are being executed. A TMS320VC33 board with a full speed USB (12 Mbps) interface already been developed. The EDL project would involve designing the high speed USB circuit using the Cypress Semiconductor USB IC CY7C68013A, its interface to the processor, and the associated USB firmware.

1. Introduction

USB has become the industry standard for external peripheral connection with computer. Older options like parallel port and RS232 for serial connection have become obsolete. USB allows the flexibility of speed based on application. A minimal speed of 1.5 Mbps for HID devices, 12 Mbps for basic data transfer (full speed) and 480 Mbps (high speed) for video applications. The need for real time computation for simulations is accomplished by the DSP TMS320vc33 but for this data to be transmitted to the computer a good throughput is necessary on the controller side. The options being IEEE1394 fire-wire or high-speed USB. Technically fire-wire can give a maximum data rate of 400 Mbps. Thus the use of high-speed USB is justified.

One of the applications for this project is the implementation of a limited closed loop control system with ADC and DAC. The control system will be implemented on the DSP.

2. Design Approach

The design required us to make two essential decisions viz.

1) Use of external FIFO for Buffering
2) 56pin or 100 pin controller.

The 56 pin controller multiplexes the port pins with the FIFO data lines and hence has lesser pins.It also consumes lesser power. The port pins on the cypress controller would however be unavailable which is acceptable. The 56 pin package was chosen as it is easily available ad we would save time in acquisition of a 100 pin cotroller. The FIFOs available as free samples are unidirectional and this would require multiplexing of the data lines between the DSP and the controller for data flow in each direction. The additional overhead would reduce the throughput. This problem can be dealt with using the internal memory of the DSP.

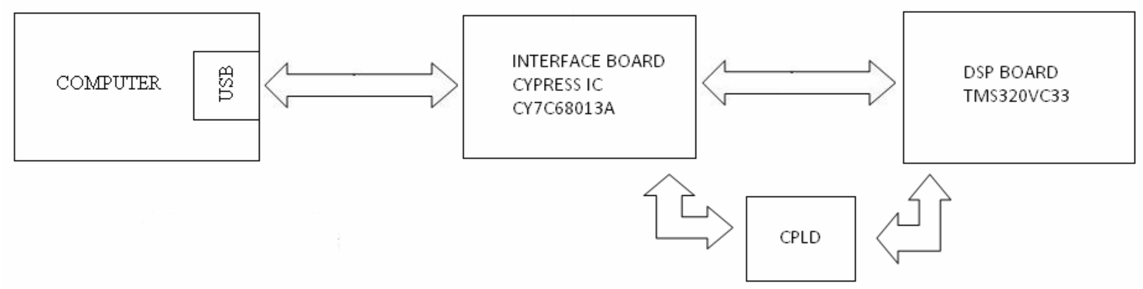

Figure 1: Block Diagram

3. Design of circuit

3.1 Power supply

The USB port supplies power at 5V. Voltage rating for CY7C68013A chip is 3.3V. Hence to convert from 5V to 3.3V TPS7333 chip is used. Power supply from the USB port can supply a maximum of 500mA as and when demanded by the devices connected to it. Since the maximum current required by our circuit is of the order of 450- 500 mA. Therefore a provision has been made either to supply power to full circuit by the USB port or power from USB port only to drive Cypress circuitry and an external power supply to drive the Digital Signal Processor circuitry including the CPLD chip. A jumper has been provided to access both the options. In all cases the USB port should be used to power the controller as the USB peripheral should be powered on plugging the device.

The external Power supply includes a diode bridge in the initial stage to rectify the a.c signal coming from the transformed-mains and is filtered by the capacitor. This is then fed to a voltage regulator (7805 LM340H-05) to get a stabilized voltage of rating 5 volts. Since the DSP chip TMS320VC33 requires the voltage supply (V_{DD}) of 3.3V and 1.8V, a circuit to convert 5V to 3.3V and 1.8V was achieved using the TPS768D318 chip and the associated circuit.

3.2 Interfacing USB controller and DSP chip

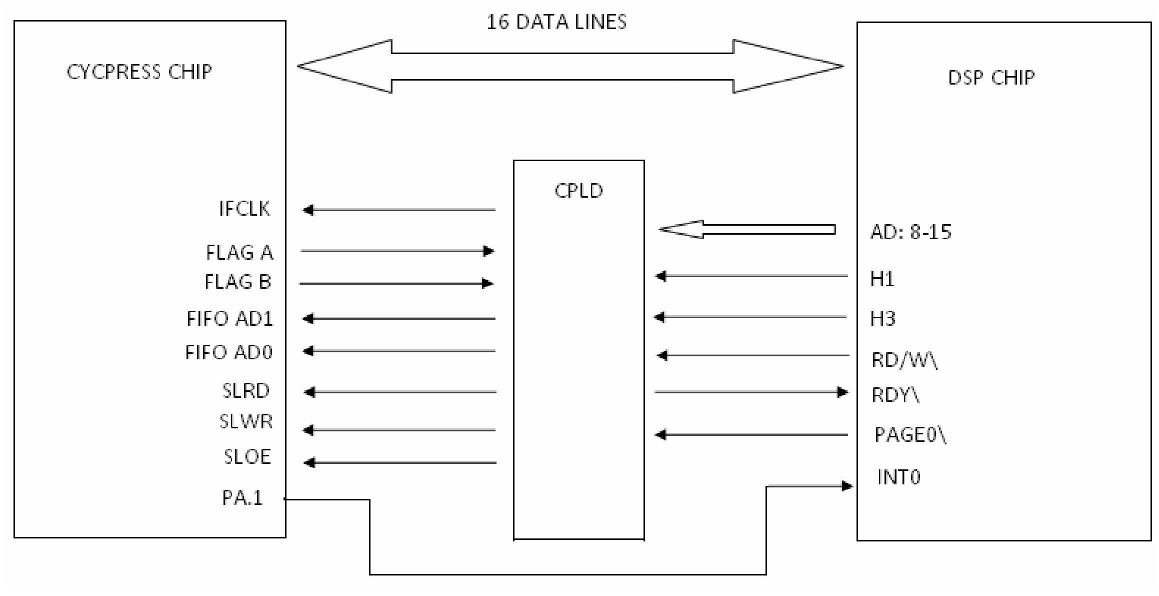

Figure 2: CPLD connections

The Endpoint buffers for the USB controller chip are implemented as FIFOs. Access to the endpoint buffers can be achieved by the CPU in two ways.

1) As Ram Blocks
2) As a FIFO

However to optimize the data transfer rates it is essential that the CPU should not interfere and hence should be accesed as a FIFO. The FIFO is directly accessible to external peripherals and can be accessed in two ways.

1) Using GPIF (general purpose interface) with the USB controller acting as the master for the transfers.
2) Using the FIFO as Slave FIFOs and the DSP acting as the master for the transfers.

The use of GPIF requires the DSP to follow cypress controller CLKOUT for synchronization. However the DSP allows external peripherals to synchronize using its own clock. Hence the USB controller FIFOs must be used as slaves.

1) Serially
2) Using the primary memory Bus
3) Using the expansion Bus

For good throughput it becomes essential to use one of the buses. The primary bus is used for ease of implementation. 16 Data lines of DSP chip are connected to 16 data lines of the controller chip (Port B and Port D) to exchange data between the two chips. The synchronizing clock is provided by DSP at 75 MHz. The slave FIFOs can handle a synchronizing clock in the range of 5-48 MHz. This makes it essential for a CPLD to divide the frequency by a factor of 2 and also correlate the handshake signals on both sides.

3.3 Digital Signal Processor TMS320VC33 circuit

Six Serial Port pins of the chip, VC33_RESET\, INT3\ and GND pins have been brought out to common pin header. This is done to connect the circuit directly to the serial port of the computer which then can be another window to the DSP for debugging. The controller chip has the ability to reset the DSP. This is particularly useful as the DSP is used in the micro-computer boot loader mode. The boot loading process begins with the reset of the DSP followed by pulling one of the four INT 0,1,2,3 pins to select the boot loading mode and address. This is achieved using the port A pin. The DSP can be reset by the power supplying chip TPS768D318. Essential jumpers are used with the CLK mode pins to enable the oscillator and select the frequency

4. Software

The software to use the DSP i.e. to issue command to the DSP and read and write to it is X-windows based. At a lower level is the command line interpreter which then forwards the data to the USB controller firmware. The controller firmware interprets the initial bytes and takes a decision based on it. The rest of the bytes are forwarded to the DSP.

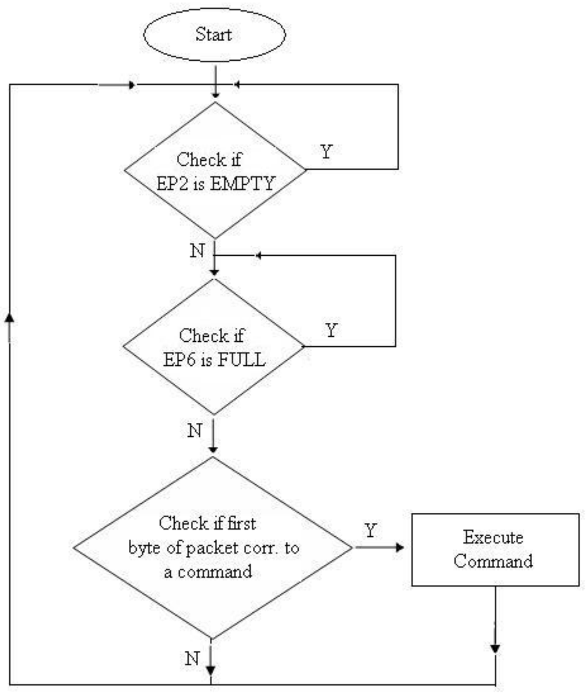

Figure 3: Flow diagram

4.1 Host side programs

The host side command line interpreter uses a series of predefined functions to carry out any given command. These functions are based on libusb functions and create interpretable instructions for the controller firmware.

The predefined functions were modified to match with the new design however maintaining the basic structure for compatibility.
For the host side programs to use libusb functions(like bulkwrite and read), a device handler has to be passed to the code. The source code for the same has been written

4.2 Cypress Firmware

The cypress firmware performs polling of the EP2CS register to check whether the out endpoint is not empty and also checks using EP6CS whether the in endpoint is not full.
When the above condition is satisfied, the polling is stopped and the command written into the firmware is interpreted. The commands written to the controller are
RESET VC33, READ _VC33 or SEND_TO_VC33.

4.3 CPLD (Complex Programmable Logic device)

The CPLD has to perform the following functions.
1) Divide the H1 clock of DSP by 2 and provide it as an input to the Controller Slave FIFO interface.
2) Generate the Read / Write signals for the USB controller (SLRD), (SLWR) considering the input signal from the DSP.
3) Generate the Ready signal for the DSP from the full/empty flags.

The following logic for the same has been implemented in VHDL.

1) IFCLK = H1/2
2) RDY/ = A or B

where A= FLAGA or (SLRD)
B= FLAGB or (SLWR)

3) SLOE=Page0/ or (not (RD/W))

4) SLRD=not (SLOE) and (IFCLK/2)

5) SLWR=Page0/ or (RD/W)

6) SLCS=0

7) PKTEND =XF0;

8) The address lines of the DSP, A8-15 are checked for addresses 0x10 and 0x11. For this the out buffer is read by generating 00 on the FIFO address lines.
For the addresses 0x12 write operation is performed & hence the in buffer is selected by generating the address 01 on the FIFO address lines.

5. Test procedures

ON receiving the board the USB connector was connected and power on the VCC pins of the DSP as well as the Cypress chip were checked.

The circuit was populated up to the USB controller. The circuit up to this point was then tested by trying to write into the controller endpoint buffers. On successful completion of the same, the board was populated further.

The working of the DSP is checked by checking the H1 and H3 clock-outs. Once this is verified the DSP is tried to be boot-loaded using the serial interface header and the full speed board. If we are able to write into DSP's memory, it indicates the proper working of the DSP.

The Working of the CPLD on a very basic level can be achieved by setting the slave FIFOs to work on external clock. If the CPLD generates the IFCLK the Cypress controller will work.

A basic DSP code that flashes its port pin XF1 was written for boot-loading into the DSP. The successful working of which confirms the working of the CPLD and the boot-loading section of the code. Also the working of the Firmware is tested by the same.

6. Further work to be done

As the USB controller works but is unable to communicate with the DSP, debugging of the board for the CPLD and DSP circuits is necessary.
A complete test of the firmware and the board with the GUI available to be done.
We need to evaluate the exact throughput of the device.
We need to add A to D converters to the board so that it can be used for real-time simulation of devices.

7. Conclusion

An attempt has been made to interface a DSP through the USB so that it can be used for real time simulation. Proper conceptualization of the design for the same was done and implemented.

Acknowledgement

We would like to express our deep sense of gratitude to Prof. M.C. Chandorkar and Prof D. K. Sharma for their inspiring guidance and valuable suggestions all through the course, which indeed made the project a great learning experience.

We would also like to thank Shiva Kumar (M.Tech student) and Melissa Fernandes (M.Tech student) for their invaluable help in various parts of the project.

References

[1] EZ-USB FX2 Technical Reference Manual

[2] TMS320C3x User's Guide

[3] TMS320VC33 and CY7C68013A Datasheets

[4] http://www.cip.physik.uni-muenchen.de/~wwieser/elec/periph/USB-FX2/software/
Author: Wolfgang Wieser, Date: 18-03-2006, LMU University, Munchen

Circuit Diagrams

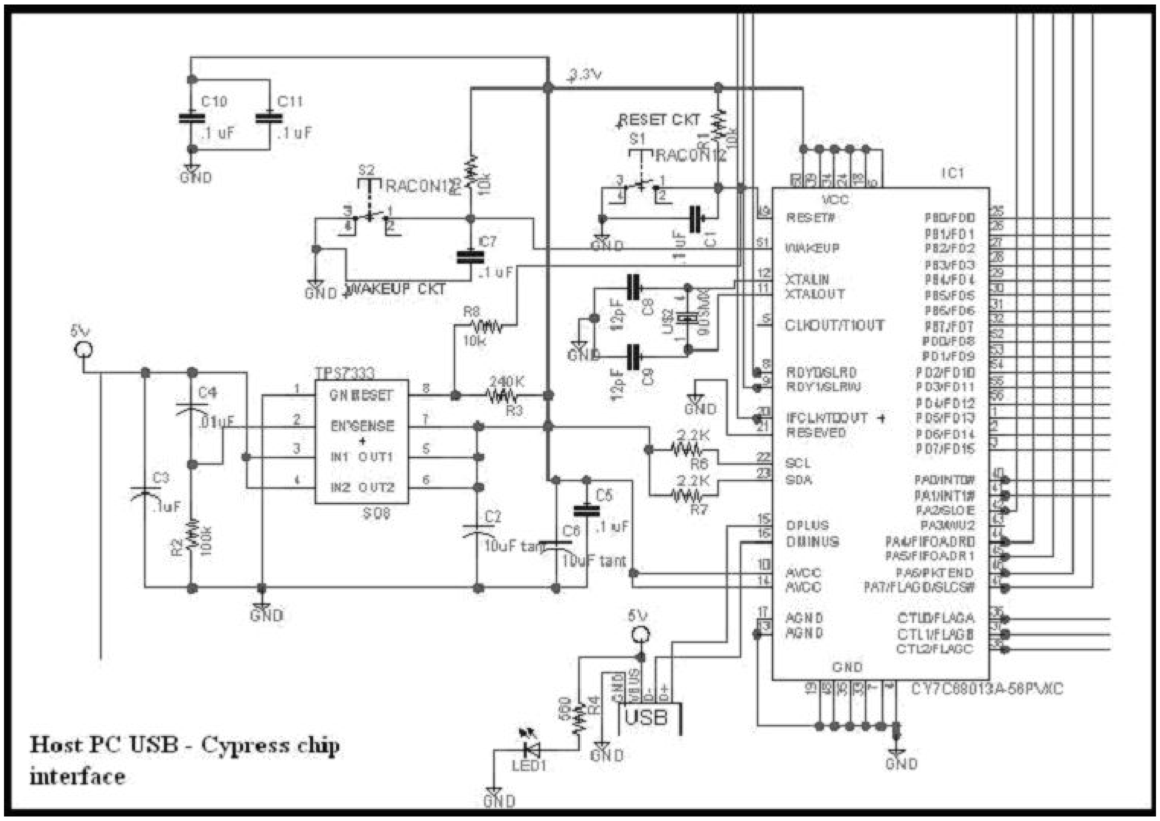

Figure 4: Cypress Controller (CYC) Chip connections

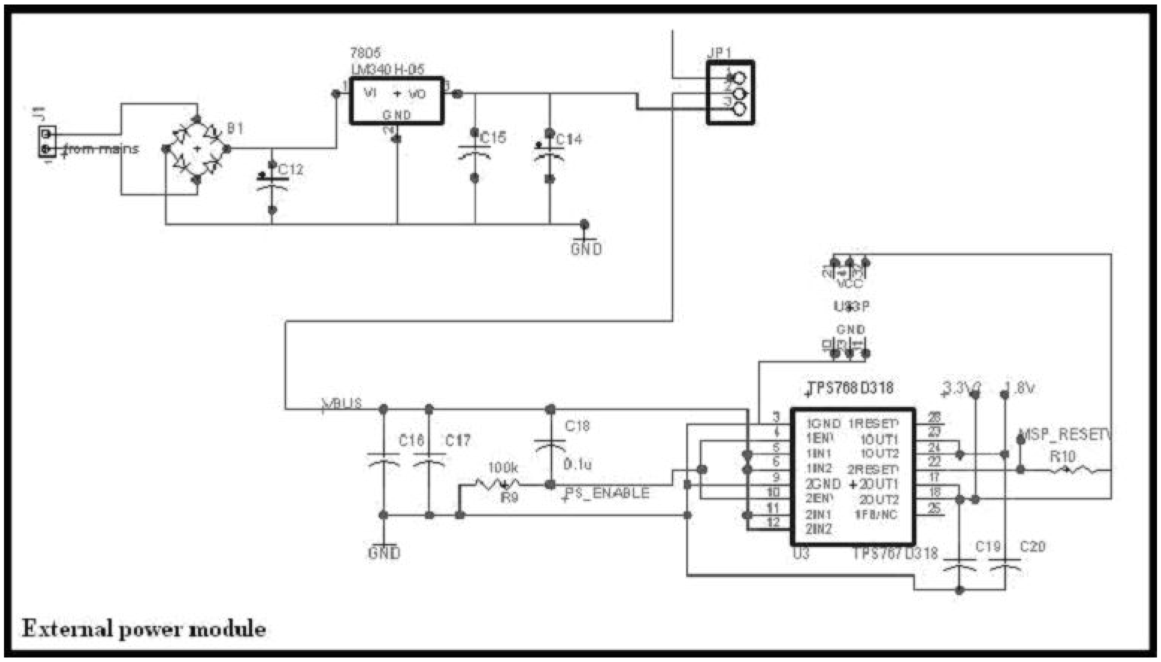

Figure 5: External Power Supply, TPS7680 (Regulator) and CPLD Power Pins connection

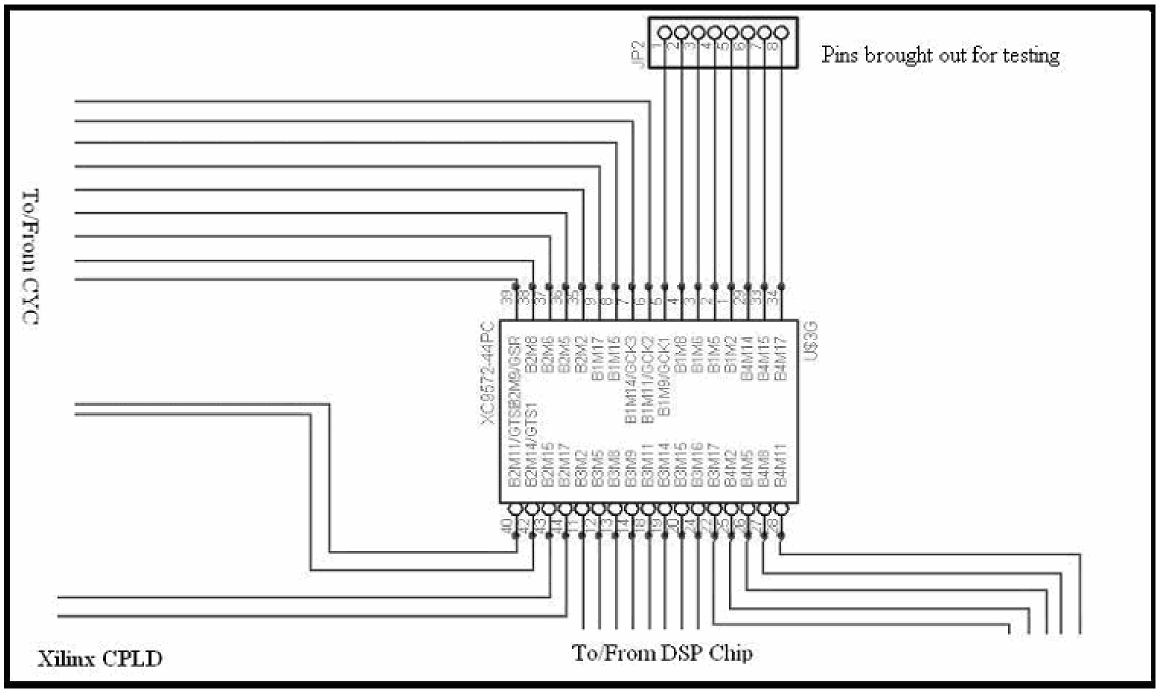

Figure 6: CPLD Connections

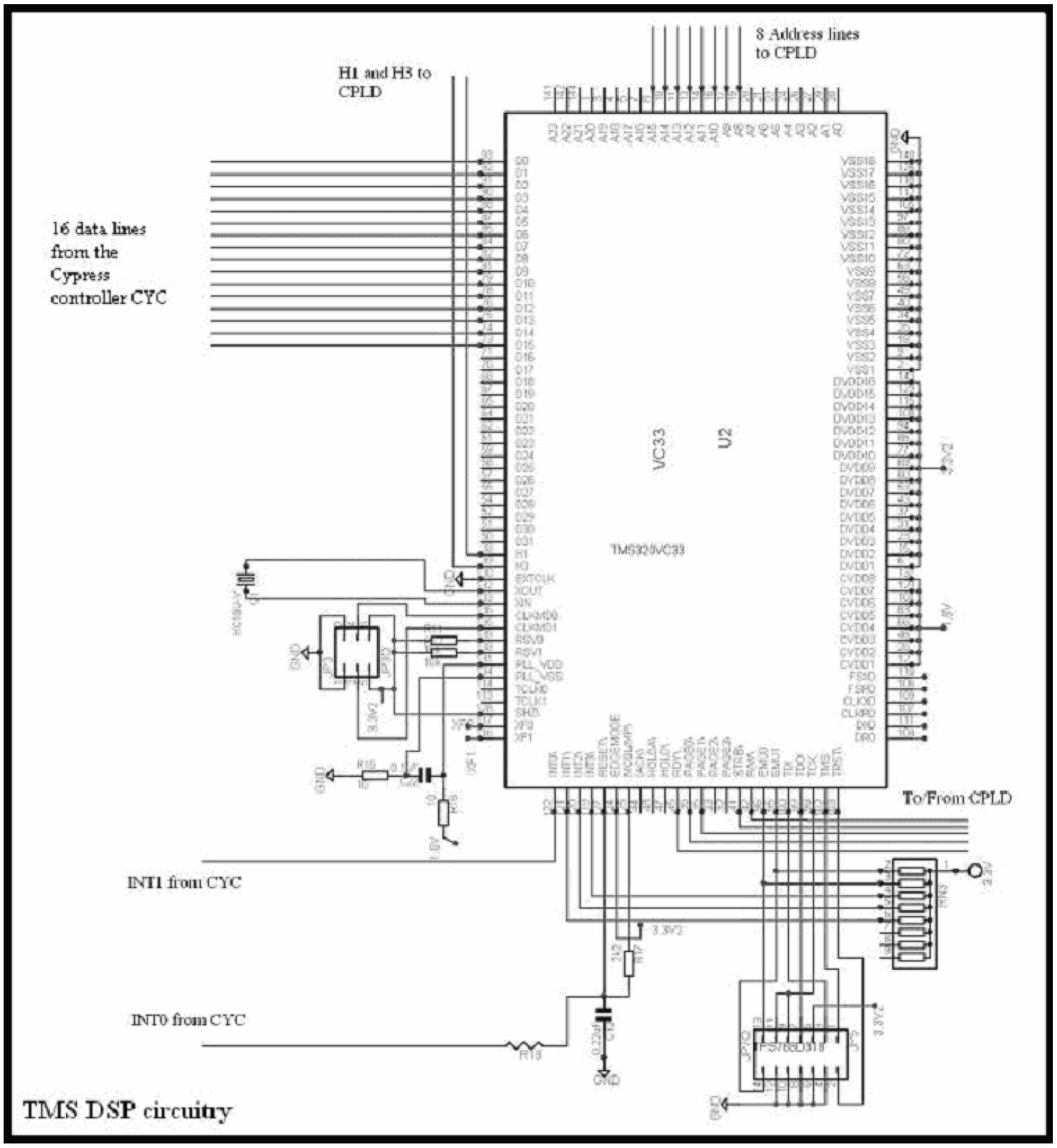

Figure 7: DSP Chip connections