EE389 Electronic Design Lab. Report

Department of Electrical Engineering, IIT Bombay, Nov 2004

Group No. D9: Supreet Joshi (01D07031), Harshad Kasture (01D07033),
Abhinav Mangal (01D07034)
Supervisor: Prof. U.B. Desai

A DSP Interface for the Chipcon RF Transceiver

Abstract

The aim of this project is to develop a DSP based board which will interface to the Chipcon CC1000 RF Transceiver. The DSP as well as the transceiver are on the same board. This board has Parallel Port Interface with a PC for booting the DSP, configuring the transceiver and sending data. The project involves building two interfaces: (i) The DSP HPI (Host Port Interface) - PC (ii) The DSP McBSP (Serial Port) - CC1000 Transceiver. The design objective is to build correctly synchronized hardware and software for both the interfaces so that the DSP operates correctly and configures the RF transceiver to operate in the right mode as specified by the host program. The PC parallel port along with necessary hardware is used to interface with the HPI port of the DSP. The product aims to be a single board "processor-cum-transceiver" which will have applications in RF ID based applications and sensor networks.

Index:

List of Tables and Figures

1. Introduction

The board is based on the TMS320C54CST DSP from Texas Instruments Inc., and the CC1000 transceiver sourced from Chipcon Inc. The DSP has an on chip bootloader ROM, a Host Port Interface (HPI) for interfacing with PC's parallel port, Multichannel Buffered Serial Ports (McBSP) for interfacing with the RF Transceiver CC1000, which is seated as a daughter card on the board. The two interfaces allow a high degree of control on the booting of DSP as well as configuration of the CC1000. The PT6932A power supply is used for power distribution, as recommended by TI.
The board needs to be used in conjunction with the software. This software is written in C++ and will run on a Windows system. It will boot up the DSP, and transfer DSP executable HEX code which will configure the Transceiver for various parameters like Transmission Frequency and data rate, etc. So, once the software is executed, the Transceiver will start functioning according to the configuration.
The DSP provides the high computational power and is capable of real time operations on the data received. In this Project the DSP boot code is transferred from a PC via the HPI interface, but storing the boot code in the onchip memory can make it an independent device capable of receiving, processing and retransmitting the data.

2. Functional Description:

2.1 Functional Diagram:

Figure 1 shows the Functional block diagram of the board. It illustrates various interfaces that exist on the board

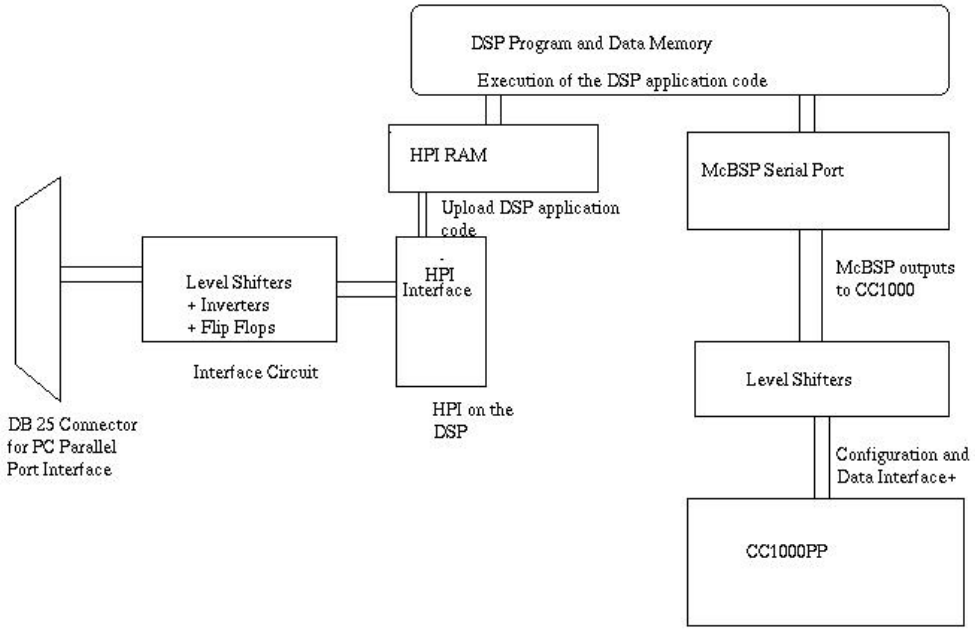


Fig. 1 Functional block diagram of the system

2.2 The DSP

The DSP has the following relevant features:

2.2.1 On Chip RAM

The entire data and program memory is on 40K * 16 byte onchip RAM, which means there are no wait states involved in memory access. This makes the memory access faster. It is also sufficient to store and execute the program for configuration of CC1000 transceiver.

2.2.2 On Chip Bootloader ROM

The C54CST includes a 128K*16 bit ROM which has a bootloader program. The bootloader can be used to automatically transfer user code from an external source to anywhere in the program memory at power up. If MP/MC of the device is sampled low during a hardware reset, execution begins at location FF80h of the on-chip ROM. This location contains a branch instruction to the start of the bootloader program. The code can be downloaded in various ways:

· Parallel from 8-bit or 16-bit-wide EPROM
· Parallel from I/O space, 8-bit or 16-bit mode
· Serial boot from serial ports, 8-bit or 16-bit mode
· UART boot mode, 8 bit
· Host port interface boot (HPI)

2.3 Host Port Interface

The HPI on the DSP is used to transfer code from the PC, as it can be interfaced easily with the Parallel Port Interface on a standard PC.

2.3.1 Features

· Sequential transfers (with autoincrement) or random-access transfers
· Host interrupt and C54x interrupt capability
· Multiple data strobes and control pins for interface flexibility

2.3.2 Basic Functionality

The HPI8 interface consists of an 8-bit bidirectional data bus and various control signals. Sixteen-bit transfers are accomplished in two parts with the HBIL input designating high or low byte. The host communicates with the HPI8 through three dedicated registers -- the HPI address register (HPIA), the HPI data register (HPID), and the HPI control register (HPIC). The HPIA and HPID registers are only accessible by the host, and the HPIC register is accessible by both the host and the 54CST.

2.4 Multichannel Buffered Serial Ports (McBSPs)

Multichannel Buffered Serial Ports are the serial interface for the DSP.

2.4.1 Features

· Full-duplex communication
· Double-buffer data registers, which allow a continuous data stream
· Independent framing and clocking for receive and transmit

2.4.2 Basic Functionality

The McBSP consists of a data path and control path. The six pins, BDX, BDR, BFSX, BFSR, BCLKX, and BCLKR, connect the control and data paths to external devices. The data is communicated to devices interfacing to the McBSP by way of the data transmit (BDX) pin for transmit and the data receive (BDR) pin for receive. The CPU or DMA reads the received data from the data receive register (DRR) and writes the data to be transmitted to the data transmit register

(DXR). Data written to the DXR is shifted out to BDX by way of the transmit shift register (XSR). Similarly, receive data on the BDR pin is shifted into the receive shift register (RSR) and copied into the receive buffer register (RBR). RBR is then copied to DRR, which can be read by the CPU or DMA. This allows internal data movement and external data communications simultaneously.

2.5 Software Description

The software is essential for the correct operation of the board. There are two components of the software:

2.5.1 DSP Code

This is the code that DSP will execute to configure the transceiver to set its various operating parameters), and send/receive data to/from the transceiver. It uses the McBSPs for communicating with the transceiver.

2.5.2 C Code

This is the code that PC will execute to transfer data to the DSP. It first establishes the HPI -PPI interface, then transfers the DSP code in HEX form to onchip RAM, and then hands control to the DSP, so that the DSP code gets executed.

2.6 The RF Transceiver CC1000

The CC1000PP is a low voltage low power transceiver with operating frequency between 300MHz to 1000MHz.

2.6.1 Configuration Interface

CC1000PP is configurable by a 3 wire serial interface, which is connected to one of the DSP McBSP. The configuration sets parameters such as:

· Receive / transmit mode
· RF output power, and output frequency
· Data rate and data format (NRZ, Manchester coded or UART interface

2.6.2 Signal Interface

The signal interface consists of 2 serial lines, the Dataline and the Dataclock. Depending on the mode selected in configuration, the data format varies. Data on the dataline is read/written on the data clock.

2.7 The Switching Regulator

The TI PT6932 A voltage regulator is used to supply the 1.5 and 3.3 volt supply to the DSP. The TI recommends this for the application boards for its DSPs. The voltage regulator provides a uniform supply of 1.5 and 3.0 volts with a 5 volt input. The features and specifications of the TI PT6932 are

- Adjustable output voltages
- Over Temperature Protection
- Soft-Start
- Line regulation for $(4.5V \leq V_{in} \leq 5.5V)$, $I_0 = I_{typ}$ is ± 7mV
- Load regulation for $V_{in} = 5V$ and $(.1 \leq I_0 \leq I_{typ})$, ± 4mV
- Vo Ripple/Noise for $V_{in} = +5V$, $Io = I_{typ}$ is 50mV

3. Detailed Specifications of Implementation

3.1 The DSP HPI – PC PPI Interface

3.1.1 The HPI Boot from HPI Bootloader

On DSP boot, the bootloader running on DSP asserts the HINT signal to check for the HPI boot. Tying the HINT output to INT2 pin on DSP ensures a HPI boot. When INT2 is activated, the bootloader assumes that the code will be loaded into on-chip RAM by an external host. The bootloader waits, polling address 007Fh of the on-chip data RAM, while the host loads its on-chip random-access memory (RAM). Once it is finished loading the on-chip RAM, the host writes the PC value into data memory location 007Fh, specifying the address of the entry point. Address 00 007Fh contains the lower 16 bits (PC) of the entry- point (address A15-A0).Note that the PC of the entry point must be a non-zero number. When the host performs the write to 00 007Fh, the C54CST detects changes at that address, and uses the new contents of the address to branch to the loaded code. Figure 2 shows the HPI Boot Flow chart.

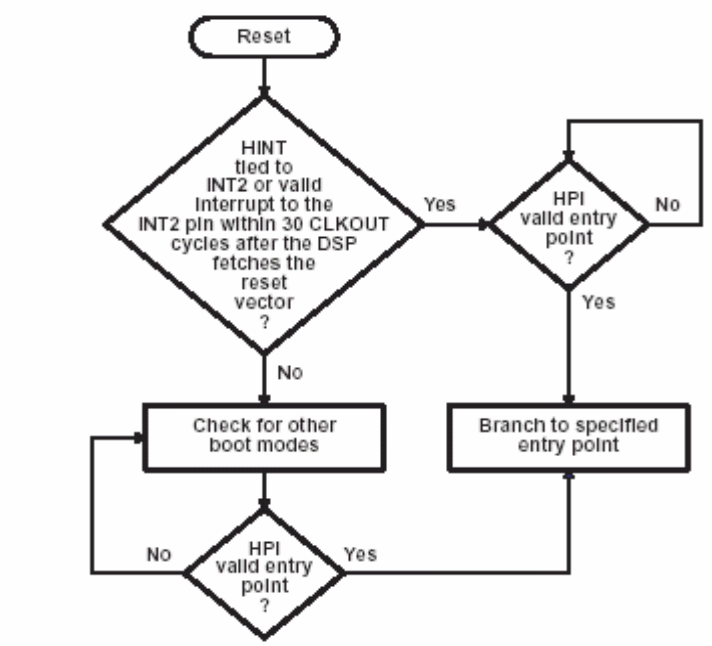


Fig. 2 HPI Boot Flow chart

3.1.2 The PC Parallel Port

The Parallel Port on a standard PC is a 25 pin interface which provides 8 bit parallel data along with status and control lines. Table 1 shows the Parallel Port signals. The outputs are TTL outputs and logic levels are also TTL levels, thus necessitating level shifters between the DSP HPI (operating at 3.3V) and the PC PPI (operating at 5V). There are 3 registers on the PC for the parallel port: Control, Data, and Status, which control the DB25 interface connector pins. The Data line is 8 bit wide, and is used to send the data to DSP. Status lines are read only, and are used for inputs from the DSP. Control lines can be read-write, but in this design they are used as output pins to DSP. The status and control lines form the handshake signals for data transmission.

PORT	R/W	I/O ADDRESS	BITS	FUNCTION
Data out	W	Base + 0	D0 – D7	Eight TTL outputs
Status In	R	Base + 1	S3 –S7	Five TTL Inputs
Control out	W	Base + 2	C0 –C3	Four TTL open collector outputs
Control out	W	Base + 2	C4	Internal, IRQ enable
Control out	W	Base + 2	C5	Internal, Tristate data (PS/2)
Data feedback	R	Base + 0		Matches data out
Control feedback	R	Base + 2	C0 –C3	Matches control out
Control feedback	R	Base + 2	C4	Internal, IRQ enable readback

Table 1 Parallel Port Signals

3.1.3 The DSP HPI

The HPI pins and their functions are as follows:

- The HPI data bus uses HD0 - HD7.
- The host interrupt, HINT, is controlled by a HINT bit in the HPIC register. HINT goes low when the DSP writes a '1' to this bit. The bit is read as a '1' by the DSP and host. If the host writes a '1' to this bit, HINT goes high and the DSP and host read this bit as a '0'. When the DSP requires the attention of the host, the DSP signals the host using this bit.
- HRDY is a DSP output indicating the DSP is ready for data transfer.
- HCNTL0 and HCNTL1 are the control signals. These signals indicate which transfer to complete. The transfer types are data, address, etc.
- HBIL low indicates the current byte is the first byte; HBIL high indicates the second byte.
- HRW high indicates the host is doing a read; HRW low indicates a write.
- There are 3 registers: HPIC, the control register, HPID, the data register and HPIA, the address register. HCNTL0/1 pins determine the register to be accessed. HRW indicates whether the access is read or write. HPIA contains the address to be accessed while HPID contains the data to be written or read.
- The data is clocked on the rising edge of HDS1.

3.1.4 The Interface

Figure 3 shows the HPI-PC interface diagram and Table 2 shows the Parallel Port pins and corresponding HPI pins.

The host accesses data by an R/W control pin, HRW, and a data strobe, HDS1. HDS2 is tied to V_{DD}. The strobe, HDS1 inputs the data at the rising edge of the signal. The PC parallel port is the host with a separate data and address bus. HAS is tied to Vdd. The falling edge of HDS1 strobes the control signals, HBIL, HCNTL0, HCNTL1, and HRW into the HPI. To permanently enable the HPI, HCS is tied to ground.

Due to an insufficient number of outputs on the parallel port, SELECTIN and AUTOFEED are combined with Flip-Flops to generate HCNTL0, HCNTL1 and HBIL, which are input to the DSP.

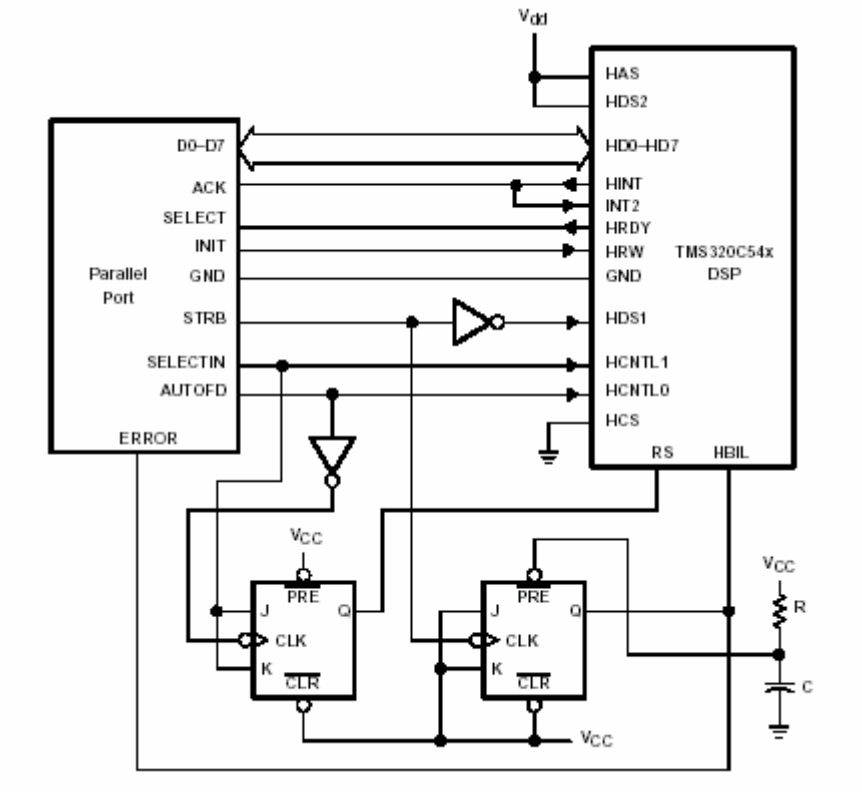


Fig. 3 HPI PC Interface

PARALLEL PORT	HPI	FUNCTION
D0 – D7	HD0 – HD7	Data bus
ACK	HINT	Host Interrupt
BUSY/PAPEREND†		No connection
SELECT	HRDY	Host ready pin
AUTOFD†	HCNTL0	Access mode control
ERROR	HBIL	HBIL status feedback
INIT	HRW	R/W control strobe
SELECTIN†	HCNTL1	Access mode control

† The parallel port generates a signal on these pins to the HPI through external hardware. HCS is set to ground. HAS and HDS are connected to V_{dd}.

Table 2 Parallel port pins to HPI

3.1.5 The timing requirements of HPI

Data is strobed into the HPI data/address register on the rising edge of HDS1. The HBIL input indicates whether the byte received is the first (low) or second (high) byte. HBIL transitions during the rising edge of HDS1, at a point later in time than HDS1. This delay is the propagation delay of the flip-flop (see Figure 4). The flip-flop ensures that the correct value of HBIL is sampled for every byte transferred. To ensure HBIL is initialized high at power up, a resistor and capacitor are connected to the flip-flop. The resistor and capacitor are not necessary if HBIL is fed back to the host for software initialization. The software

monitors the HBIL input to ensure the validity of a data/address transfer. The timing diagram of Figure 4 shows the control signals and data strobed in and out of the HPI. HAD, HD Read and HD Write are the read and write data signals.[1] The signals show the data transferred across the HPI data bus (HD0HD7).

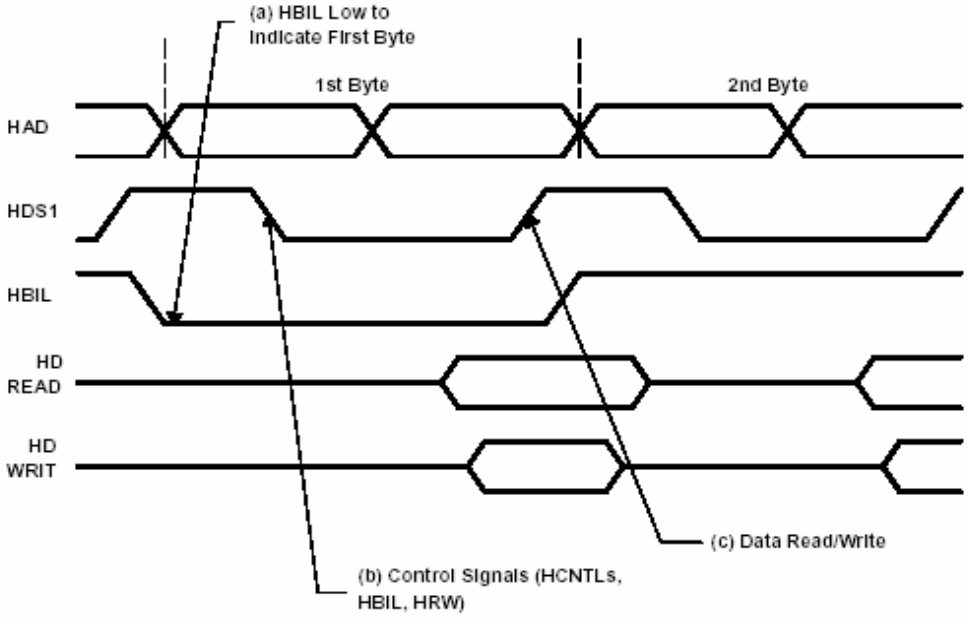


Fig. 4 HPI timing diagram

At time (a), the control signals, HCNTL1, HCNTL0, HRW, and HBIL are set to the required logic level. This level indicates the type of access necessary, whether read or write, and the appropriate registers.
At time (b), the falling edge of HDS1 causes the control signals to latch into the HPI and set the HPI to the required mode.
At time (c), the rising edge of HDS1 indicates data is being written to the HPI or read from the HPI.

3.1.6 Software Design

The software design for HPI – PC interface essentially handles the DSP boot after it comes out of reset. There are 2 designs possible:

1. With a Kernel Code: Kernel program would first be loaded, and then this code would have established the HPI interface and further operations. Essentially: DSP Reset --> HPI Boot --> Only Kernel Load --> DSP branches to Kernel --> Kernel executes --> Set up HPI --> Load the Application Code into Program memory --> Branch from Kernel to Application Code. Such a design is useful when the application code can change for different applications.
2. Without Kernel Code: Since the C54CST is already out of reset when HPI transfer begins, we directly transfer the application code just after booting. This design was chosen as it is faster, and since the board is a dedicated application intended product, the application code will not change too much. Avoiding the kernel phase allows faster transfer as the HPI operates in Host-Only-Mode, without any restriction on data rate on the DSP side.
   See Appendix A for detailed operation of the software.

3.2 The DSP McBSP – CC1000 Interface

3.2.1 CC1000 Configuration

CC1000 is configured via a simple 3-wire interface (PDATA, PCLK and PALE). There are 36 8-bit configuration registers, each addressed by a 7-bit address. A Read/Write bit initiates a read or write operation. A full configuration of CC1000 requires sending 29 data frames of 16 bits each (7 address bits, R/W bit and 8 data bits).

In each write-cycle 16 bits are sent on the PDATA-line. The seven most significant bits of each data frame (A6:0) are the address-bits. A6 is the MSB (Most Significant Bit) of the address and is sent as the first bit. The next bit is the R/W bit (high for write, low for read). During address and R/W bit transfer the PALE (Program Address Latch Enable) must be kept low. The 8 data-bits are then transferred (D7:0). See Figure 5.

The configuration registers can be read back by the DSP by setting the R/W bit low and reading the PDATA line.

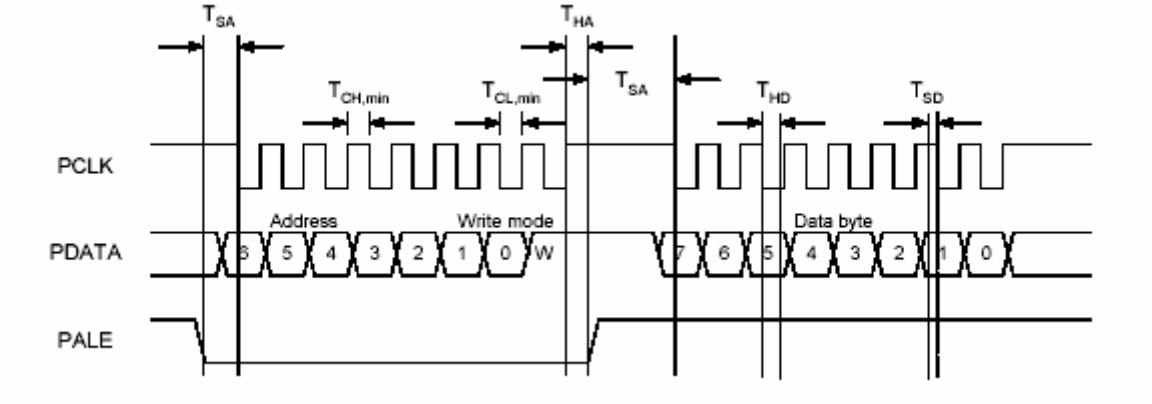


Fig. 5 CC1000 Configuration timing

3.2.2 CC1000 Data Transfer

The signal interface consists of DIO and DCLK and is used for the data to be transmitted and data received. DIO is the bi-directional data line and DCLK provides a synchronous clock both during data transmission and data reception. There are 3 different data formats

3.2.3 The DSP McBSP

The two multi channel buffered serial ports, McBSP0 and McBSP1 of the DSP are used for serial communication with the CC1000 RF transceiver. Each McBSP has three external data and control pins: CLKR/CLKX (receive/transmit clock), FSR/FSX (receive/transmit frame synchronization pulse) and DR/DX (receive/transmit data). McBSP0 is used for the programming interface of the CC1000 while the McBSP1 is used for the data interface. The data to be serially transmitted from the serial port is written in an internal 16-bit transmit data register (XDR), while the received data is read from the 16-bit internal receive data register (RDR). For the programming interface the BSP clock is set as input as the programming clock (PCLK) is provided by the CC1000. This clock is used both for data transmission and reception. For the data interface the BSP clock is set as output. The internal clock rate generator output is used for both the transmit and the receive clocks.

3.2.4 DSP Application Code

The application code that is to be run on the DSP for configuration of CC1000 and data transmission is first compiled in C and then converted to hex using the COFF2HEX utility available from Texas Instruments Inc. This code has the following functions:

- Configure the McBSPs for proper operation as required by the interface.
- Configure the CC1000 for right data format and data rate, among other parameters.
- Send the data to be transmitted on the signal serial port (DIO), or receive the data sent on the DIO serial line.

3.2.5 The McBSP – CC1000 interface

The interface between the DSP and CC1000 is shown in the Figure 6. The signals on the McBSP of DSP are generated to meet the timing specifications of CC1000 serial lines.

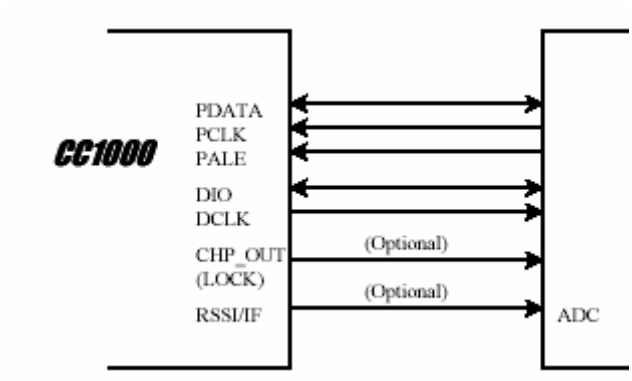


Fig. 6 CC1000 – DSP Interface

3.2.1 DSP Application Code

The application code that is to be run on the DSP for configuration of CC1000 and data transmission is first compiled in C and then converted to hex using the COFF2HEX utility available from Texas Instruments Inc. This code has the following functions:

- Configure the McBSPs for proper operation as required by the interface.
- Configure the CC1000 for right data format and data rate, among other parameters.
- Send the data to be transmitted on the signal serial port (DIO), or receive the data sent on the DIO serial line.

3.3 Hardware Development

The following issues were encountered and solved in hardware development:

- The PCB involves both digital and analog circuits so there is a possibility of noise introduction into the analog signal due to the high frequency digital

switching. Therefore Ground plane was provided to absorb the switching noise.

- The Pad clearance for the DSP package (144 pin PGE) is 5 mili inches therefore the minimum clearance setting was kept as 5 mil in the layout specifications in EAGLE. The wires in the rest of the circuit were separated manually for a better clearance.
- The minimum signal line width for the wires connecting the DSP was kept 10 mil. while it was kept 16 mil in rest of the board. The 5volt, 3volt and ground signals are kept 24mils but this could not be followed for the wires connecting the DSP and shifter SMDs. The minimum diameter for the vias is kept 24mils in order to allow close routing in the DSP area.
- The Antenna needs an impedance matching in to prevent the power loss of the signal. Therefore the antenna is kept at the minimum possible physical distance with the transceiver daughter card and the signal track length is kept as small as possible.

4 Conclusion

The final board is tested and the PC-DSP interface is working properly. The output was shown at one of the multipurpose I/O pins of the DSP. The Transceiver-DSP interface working could not be achieved. Another possible improvement in the design could be including an analog input port for the DSP to accept analog data from devices like video camera.

Appendix A: Detailed Operation of Software

This appendix details the operation of the C software.

- Signals involved :
  1. HINT input => read ACK status line
  2. HRDY input => read SELECT status line
  3. HRW output => write on the INIT control line
  4. HDS1 output => rising edge on STROBE control line
  5. HCNTL0/1 output => write AUTOFEED / SELECT control lines
  6. HBIL input => read ERROR status line

No pull up resistors are required as all read lines are from Select register and all writes to Control Reg. A read on Control Register would have required pull up resistors.

On Reset, DSP will assert HINTbar Output pin High. So, ACK will be high. HINT bit is read as zero when HINTbar pin is high, i.e. HINT=0. So, check for ACK high and read HPIC for HINT =0. This will confirm that DSP is in reset. So, bring it out of reset by writing HINT 1 in HPIC.

- Check if ACK is high (indicates start of HPI Boot)
- Select HCNTL0/1 to 01 to select HPIC
- Read HPIC HINT bit
- Check if HINT is 0 => DSP in Reset.
- Write HINT=1 in HPIC to bring it out of reset.

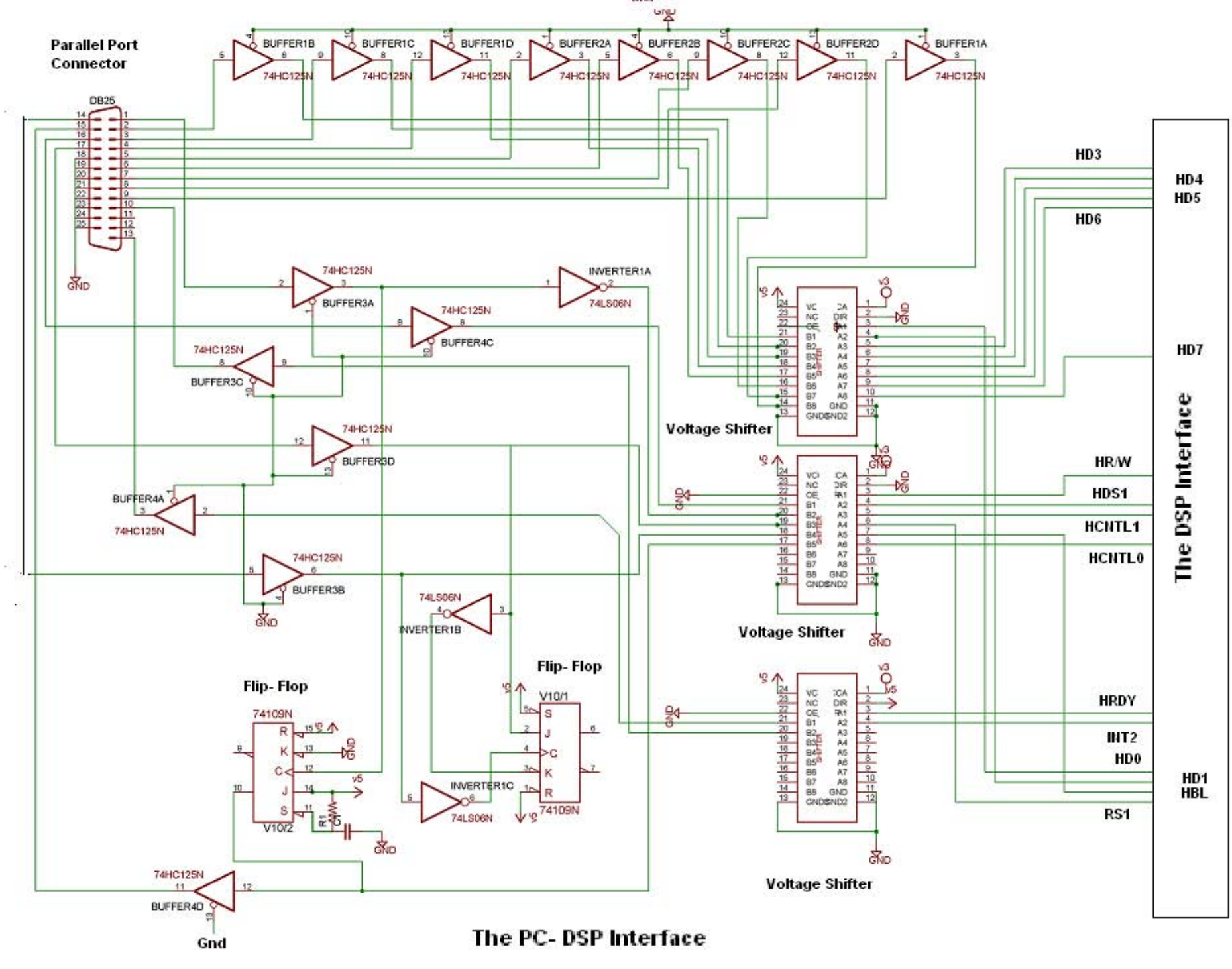
Set control word:

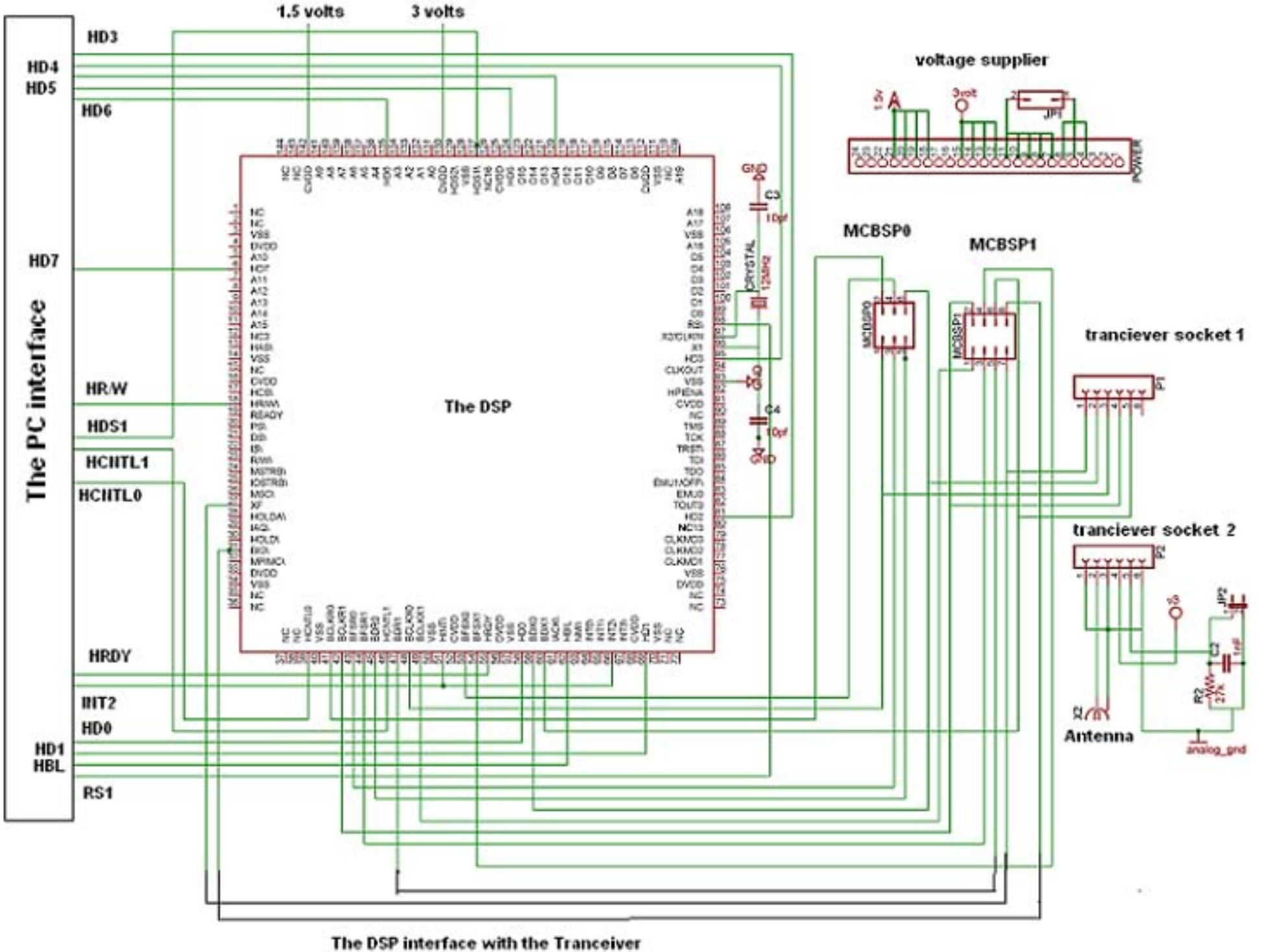
- HCNTL0 =0 and HCNTL=1 (Select HPIC)

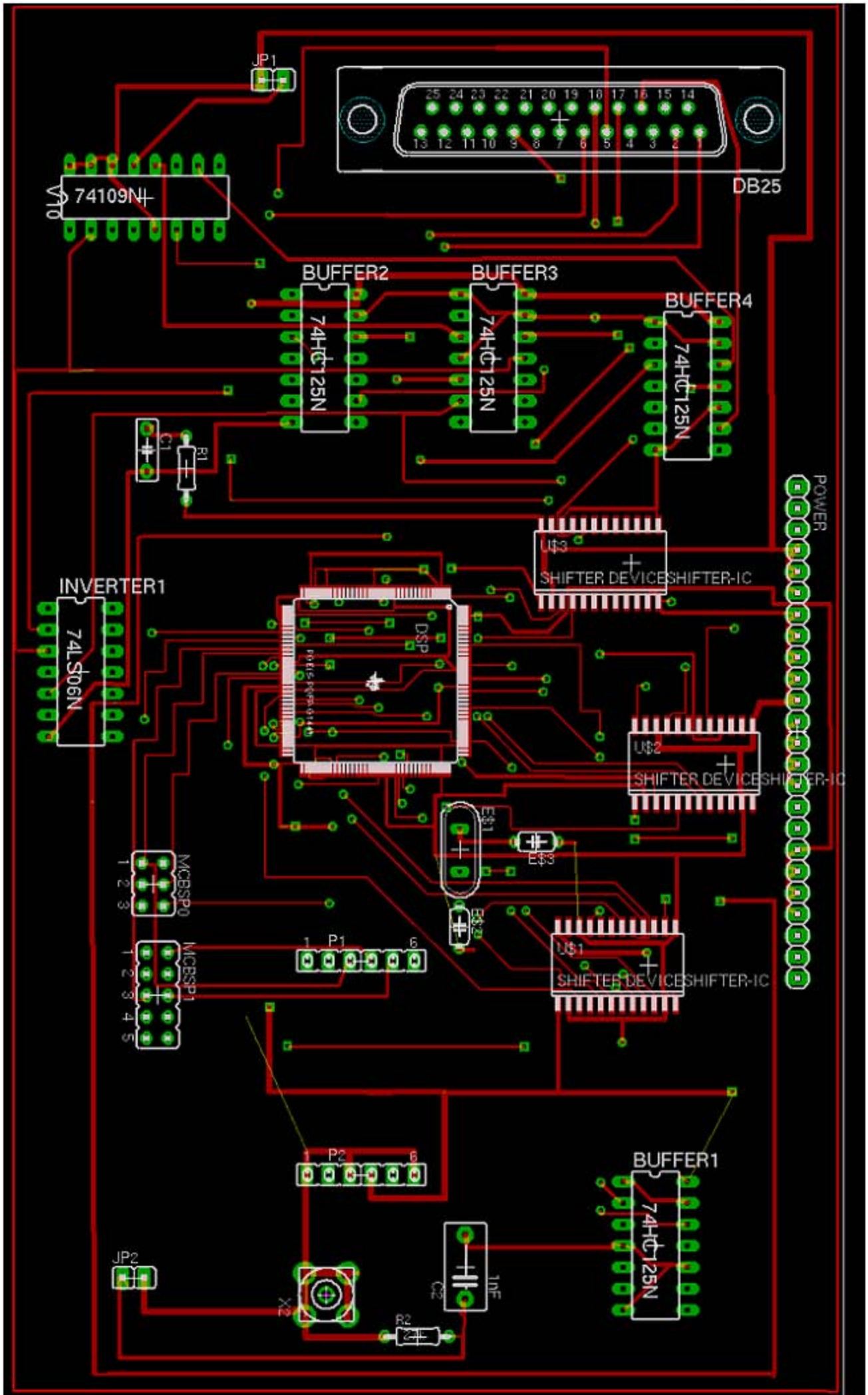
- Read HPIC SMOD and HINT : check for SMOD=1 and HINT=0 .
- Write 1 to BOB bit (Byte Order : LSB first)
- Both bytes must be same for control word write.

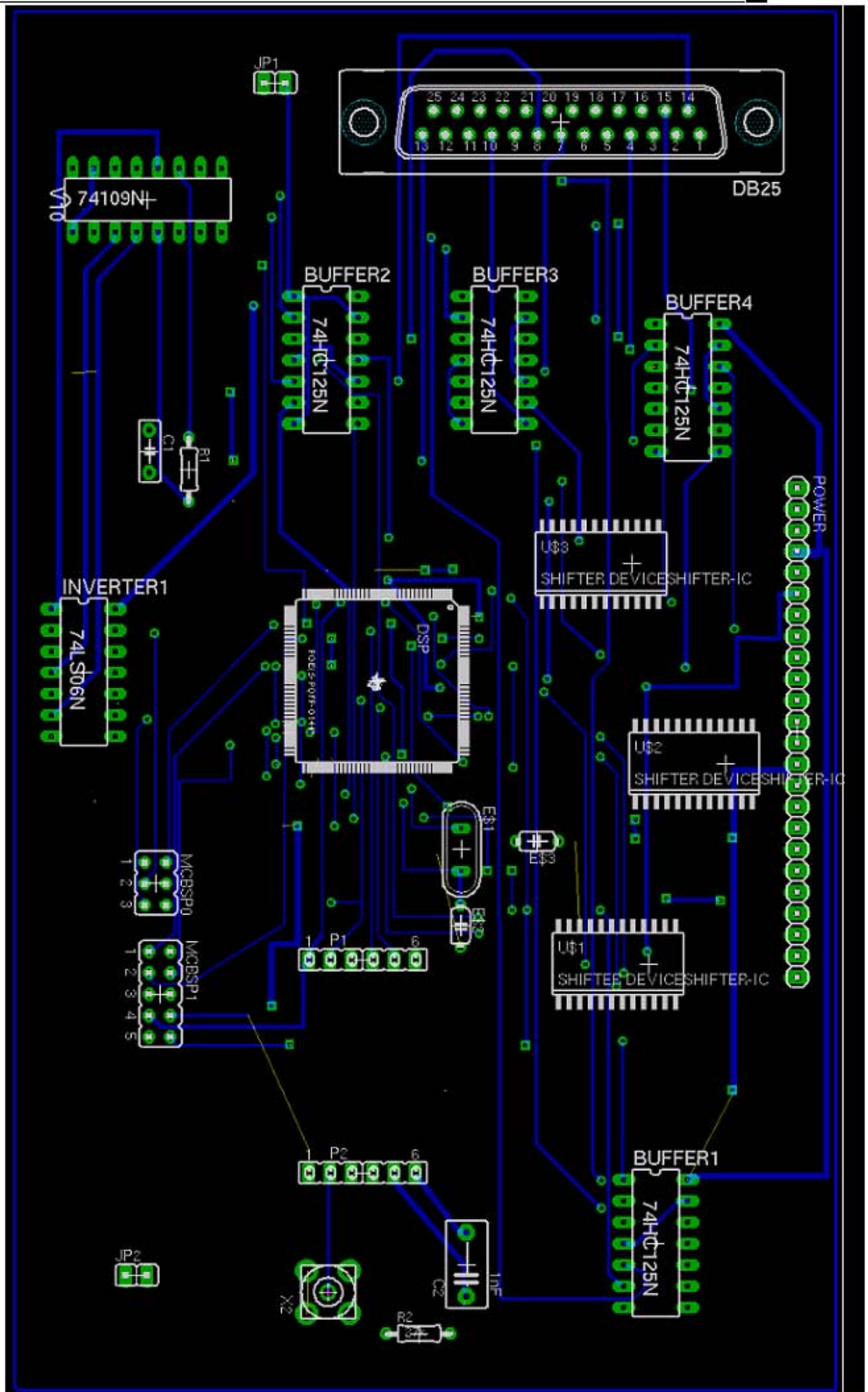
First HDS high (STROBE High) will put HBIL=0 indicating Lower Byte to be transferred (RC circuit ensures that HBIL is 1 on reset).
Finally set 007F to the start address of the DSP application code (DSPADD).

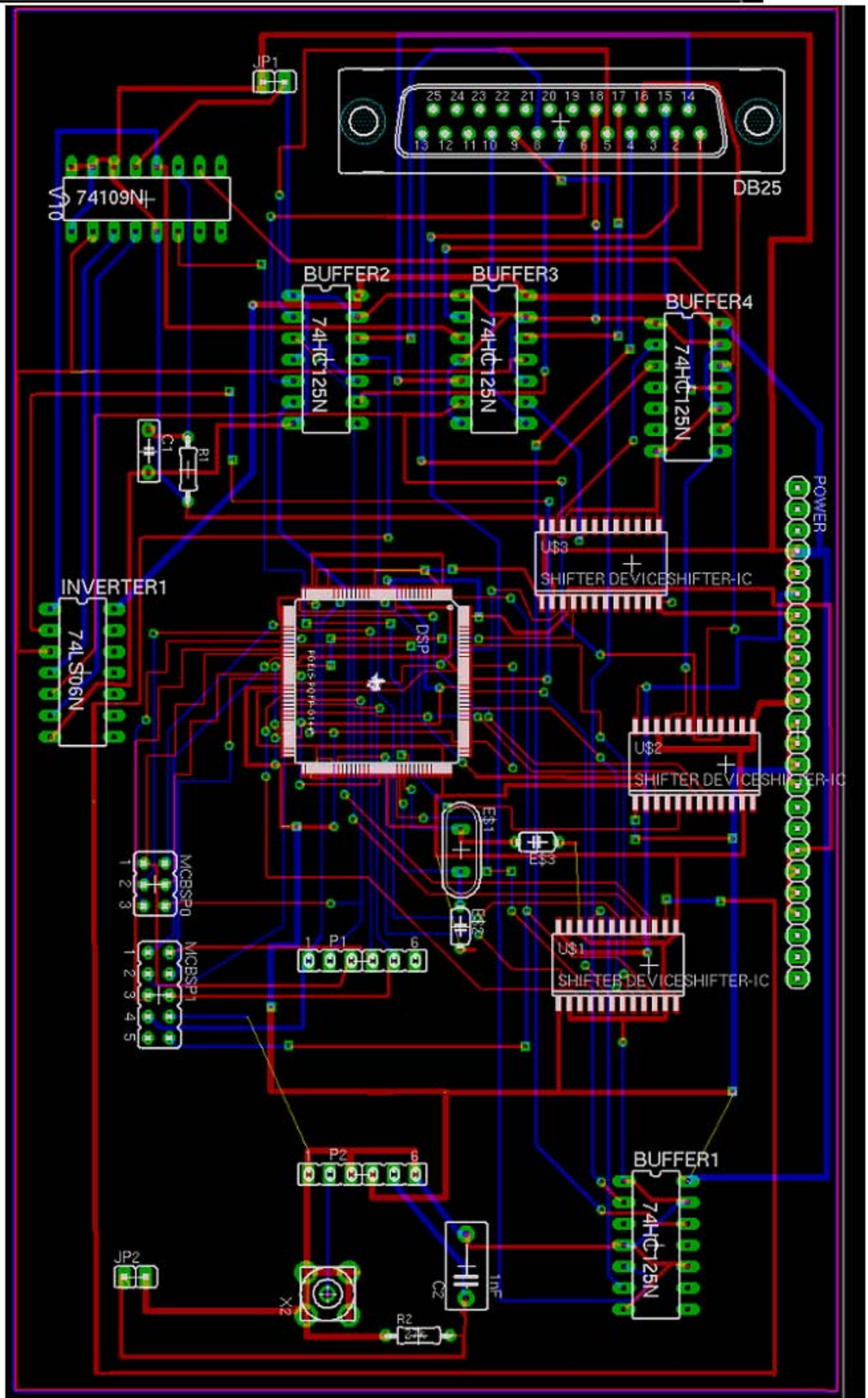
- select HCNTL0/1 to 10 to select HPIA
- select HCNTL0/1 to 11 to select HPID
- write DSPADD onto Data Bus


Parallel Port Connector
DB25
BUFFER1B
BUFFER1C
BUFFER1D
BUFFER2A
BUFFER2B
BUFFER2C
BUFFER2D
BUFFER1A
74HC125N
BUFFER3A
BUFFER4C
BUFFER3C
BUFFER3D
BUFFER4A
BUFFER3B
BUFFER4D
INVERTER1A
74LS06N
INVERTER1B
INVERTER1C
Voltage Shifter
Flip- Flop
74109N
V10/1
V10/2
R1
C1
GND
Gnd
HD3
HD4
HD5
HD6
HD7
HR/W
HDS1
HCNTL1
HCNTL0
HRDY
INT2
HD0
HD1
HBL
RS1
The DSP Interface
The PC- DSP Interface


1.5 volts
3 volts
The PC interface
HD3
HD4
HD5
HD6
HD7
HR/W
HDS1
HCNTL1
HCNTL0
HRDY
HINT2
HD0
HD1
HBL
RS1
The DSP
voltage supplier
MCBSP0
MCBSP1
tranciever socket 1
tranciever socket 2
Antenna
analog_gnd
The DSP interface with the Tranceiver


JP1
DB25
74109N
BUFFER2
BUFFER3
BUFFER4
74HC125N
C1
R1
POWER
U$3
SHIFTER DEVICESHIFTER-IC
INVERTER1
74LS06N
DSP
U$2
E$1
E$3
E$2
MCBSP0
MCBSP1
P1
U$1
P2
BUFFER1
JP2
X2
1nF
C2
R2


JP1
25 24 23 22 21 20 19 18 17 16 15 14
13 12 11 10 9 8 7 6 5 4 3 2 1
DB25
V10
74109N
BUFFER2
74HC125N
BUFFER3
74HC125N
BUFFER4
74HC125N
C1
R1
POWER
U$3
SHIFTER DEVICESHIFTER-IC
INVERTER1
74LS06N
DSP
U$2
SHIFTER DEVICESHIFTER-IC
E$1
E$3
E$2
MCBSP0
1 2 3
MCBSP1
1 2 3 4 5
P1
1
6
U$1
SHIFTER DEVICESHIFTER-IC
P2
1
6
BUFFER1
74HC125N
JP2
X2
C2
1nF
R2


JP1
25 24 23 22 21 20 19 18 17 16 15 14
13 12 11 10 9 8 7 6 5 4 3 2 1
DB25
VR10
74109N
BUFFER2
74HC125N
BUFFER3
74HC125N
BUFFER4
74HC125N
C1
R1
POWER
U$3
SHIFTER DEVICESHIFTER-IC
INVERTER1
74LS06N
DSP
U$2
SHIFTER DEVICESHIFTER-IC
E$1
E$3
R$2
MCBSP0
MCBSP1
P1
U$1
SHIFTER DEVICESHIFTER-IC
P2
BUFFER1
74HC125N
JP2
X2
C2
1nF
R2