University of California at Berkeley

College of Engineering

Department of Electrical Engineering and Computer Science

EECS 150 — Mike Lowey

Fall 2001 — Yuh Meei Seah

Project Checkpoint 1

Keypad and Display Interface

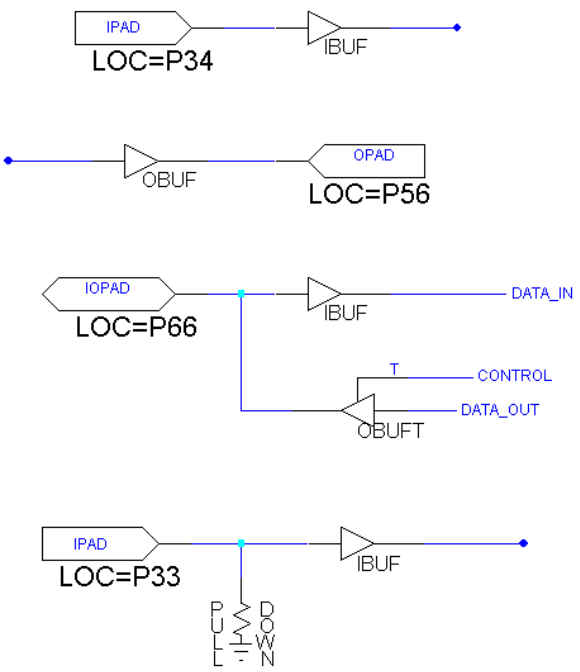
IO BASICS:

IO Block

In order to input and output signals, there are several fundamental components necessary: IBUF and IPAD for input, OBUF and OPAD for output, IBUF and OBUFT and IOPAD for two-way signals.

All of the pads (IPAD, OPAD, IOPAD) are used to make connections to pins. They must be configured by right clicking a pad and editing the symbol properties. A "LOC" parameter must be added, and its value should be "P41," for a connection to pin 41, or "P36" for a connection to pin 36, etc.

In addition to pads, each pad should have a corresponding buffer (IBUF, OBUF, or OBUFT). Connections should be made like this:



Notice that the OBUFT is a tri-state buffer. Be aware that the control signal "T" is active low, so the output buffer will be enabled (i.e. signals will be output to the pad) when T = 0. When the IOPAD is not being used to output signals, the OBUFT should be turned off (to Z state).

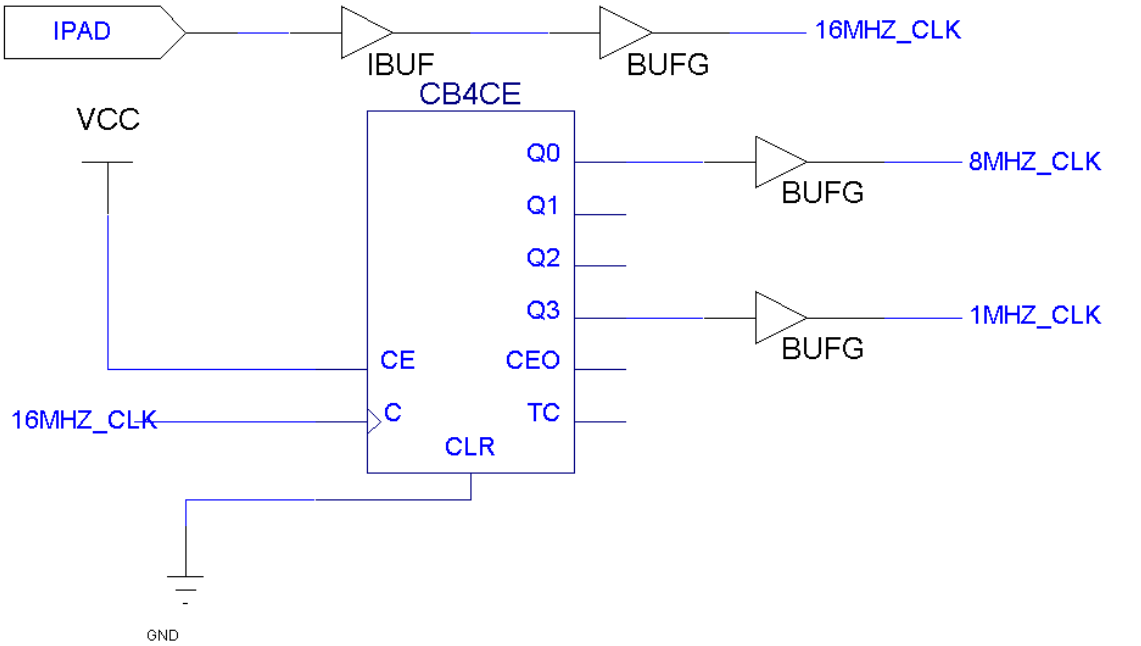
The pull-down resistor is used to dissipate charge that can build up charge on some input pins that can lead to false high readings.

Clock Divider

There is an onboard clock that runs at 16mhz, which is hardwired to pin 13 (an IPAD should be used to connect to this clock signal). A counter can be used to divide the clock in divisions of multiples of two. By using the 16mhz clock as the clock input for a counter and always enabling the counter, the LSB of the counter will be a 8mhz clock, the next bit will be a 4mhz clock, etc.

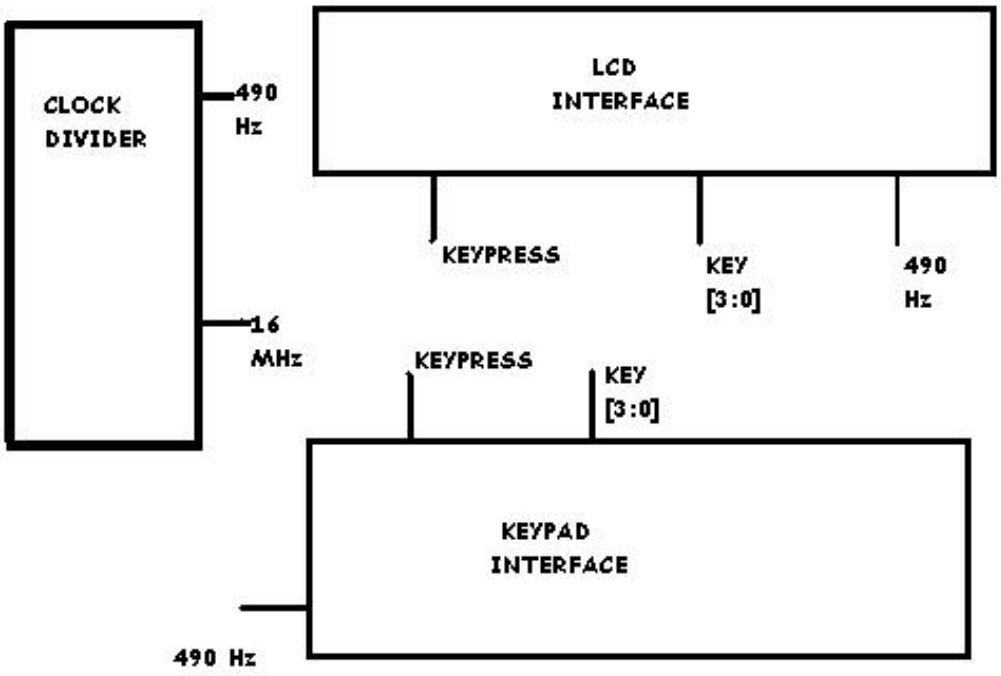
When generating clock signals, it is VERY IMPORTANT to put a BUFG before sending the clocks to components. This will prevent clock skew, which happens when different clocks become unsynchronized.

For this checkpoint we will need a 490Hz clock, but in the future you may wish to add another clock frequency. Just remember to use a BUFG for each clock.

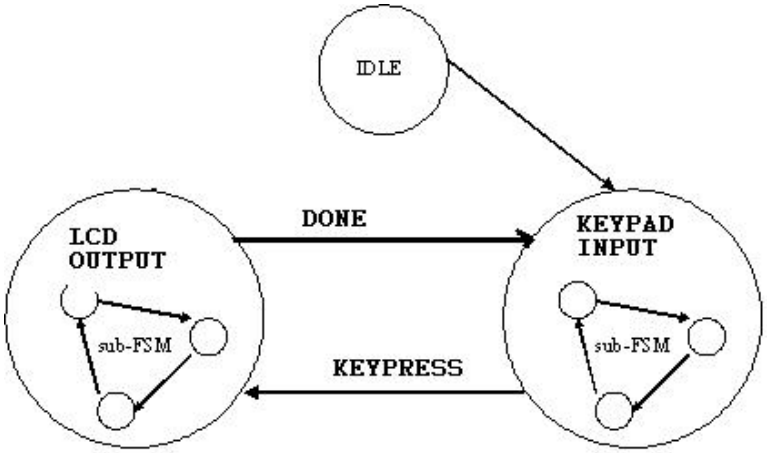


CHECKPOINT 1 OVERVIEW:

In the first checkpoint the task will be to interface the 4x4 keypad to trigger text to be written to the 2x16 character LCD. To do this you will need to create the three basic blocks in the top level diagram below:

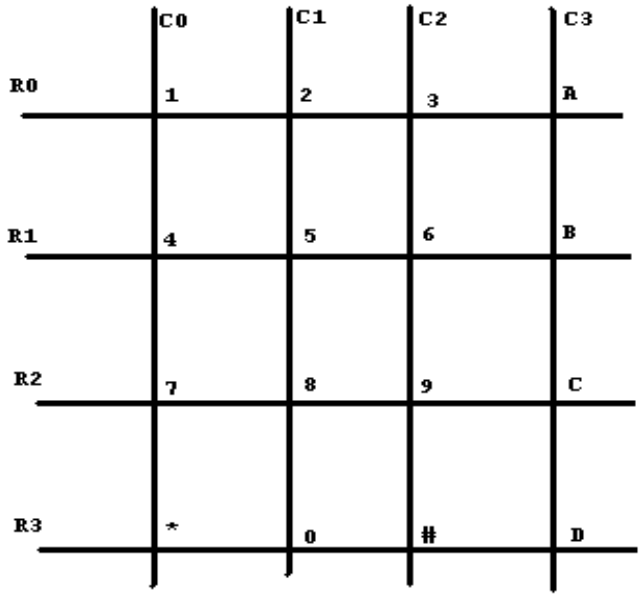


The top level diagram suggests the following top level state machine:



THE KEYPAD:

The keypad is a 4X4 grid. When a button is pressed a vertical wire is connected to a horizontal wire. The buttons are encoded in the pattern below:



The pins at the bottom of the keypad correspond to the following columns / rows:

PIN #	7	6	5	4	3	2	1	0
	R1	R2	C0	R3	C1	C2	C3	R0

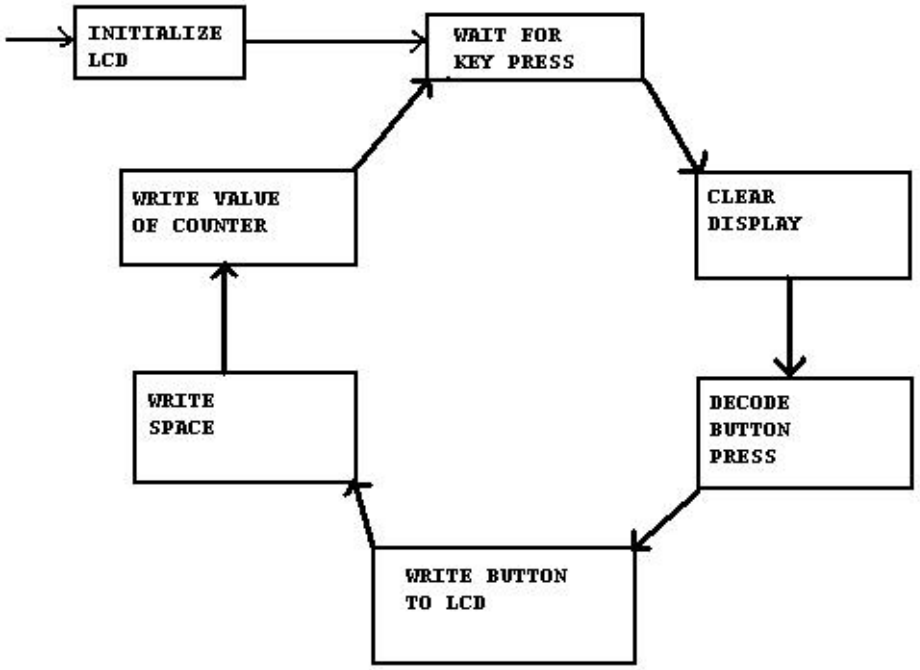
You will need to read the keys in continuously and update the KEY[3:0] output when there is a new key pressed. KEY[3:0] keeps the position of the last button pressed. The signal KEYPRESS will also be pulsed when a new key is pressed as well. The keypad interface should use pull-down resistors on the pins that are used to read from the key-pad.

The following pins should be used for the keypad:

XILINX PIN	ROW / COLUMN
71	C3
80	C2
81	C1
82	C0
83	R3
84	R2
67	R1
70	R0

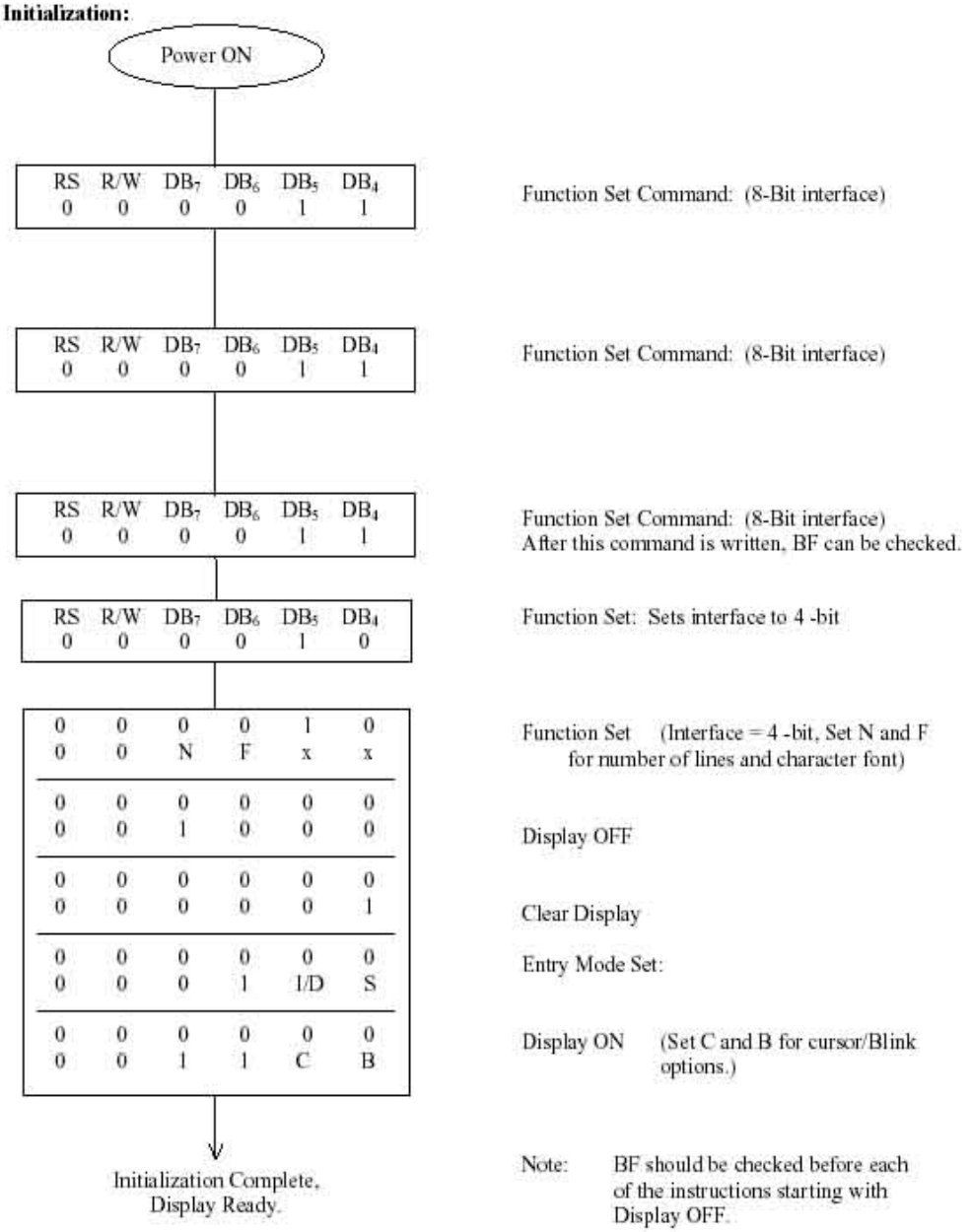
THE LCD CONTROLLER:

Now that the easy part is over, interfacing the LCD should give you more opportunities for creativity. There is extensive documentation for the LCD and its controller chip in the DATA_SHEETS file in the CS150 directory. In the project we are going to use the LCD to display the current action of the CD player. We would also like it to display time information. In check point one we will be creating capabilities to be used later in the project. We will use the keypad to trigger different outputs to the screen that uniquely correspond to each key. To simulate time, we will have a counter running continuously and output its value to the LCD each time the key is pressed as well. The behavior of the LCD interface should correspond to the following state machine:



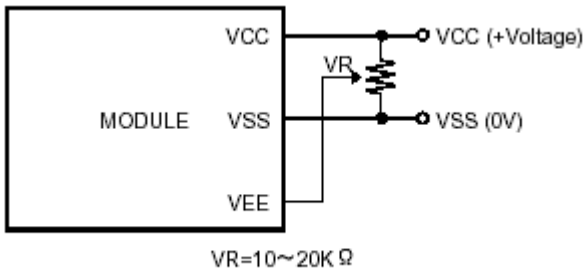
We will be using the LCD in its 4-bit mode to save pins. The 4-bit mode requires that we send our 8-bit word to the LCD in two parts. We will send the upper half of the byte (bits 4-7) first, followed by the lower half. The first step is to initialize the LCD immediately after loading your bit-file. The power-up and initialization procedure is described in detail in the data sheet named OPTREX in the DATA_SHEETS folder. The initialization procedure that we will be using is outlined on the next page. Being generous and kind, we are going to give you the clock frequency that works: 490 Hz (actually 488.28). Data can be written to the LCD 490 times a second. The display control commands are on page 35 of the OPTREX data sheet. This data sheet is fairly short and explains the capabilities of the LCD well. Please read it.

2.2.2.2 4 - Bit Initialization:



One instruction is written every 488 Hz.

It will be necessary to use a potentiometer to adjust the contrast of the LCD. IT should be hooked up as shown:



The pins on the LCD can be decoded with this handy chart:

No.	Symbol	Level	Function
1	Vss	—	Power Supply (0V, GND)
2	Vcc	—	Power Supply for Logic
3	VEE	—	Power Supply for LCD Drive
4	RS	H / L	Register Select Signal
5	R/W	H / L	Read/Write Select Signal H : Read L : Write
6	E	H→L	Enable Signal (No pull-up Resister)
7	DB0	H / L	Data Bus Line / Non-connection at 4-bit operation
8	DB1	H / L	Data Bus Line / Non-connection at 4-bit operation
9	DB2	H / L	Data Bus Line / Non-connection at 4-bit operation
10	DB3	H / L	Data Bus Line / Non-connection at 4-bit operation
11	DB4	H / L	Data Bus Line
12	DB5	H / L	Data Bus Line
13	DB6	H / L	Data Bus Line
14	DB7	H / L	Data Bus Line

Because we are using 4-bit addressing, we will not be using pins 7-10. They can be left unconnected. The remaining pins should be connected to the following Xilinx pins:

XILINX PIN NUMBER	**LCD SIGNAL**
60	DB7
59	DB6
58	DB5
57	DB4
66	E
65	R/W
72	RS

The output on the LCD should be in the following form:

Key pressed:	LCD output:
1	1 t3 t2 : t1 t0
2	2 t3 t2 : t1 t0
*	* t3 t2 : t1 t0
A	A t3 t2 : t1 t0

Where t[3:0] are the values of the output of counter when the button is pressed set up to look like time.

A working bit-file named CHECK1.BIT will be in the CS150 directory. You can use this to test your wire wrap job and contrast adjustment.

PRELAB PREPARATION:

There are two main tasks that should be completed before you come to lab. The key-pad and the LCD should be wire-wrapped to the board using the specified pins. You should wire wrap both of them to the two rows of pins that are closest to the Xilinx chip to leave room for the parts that need to be added later in the project. The second task is to have detailed FSMs worked out for the design of the keypad and LCD interfaces. Space is going to be tight for this project and it is important that you consider how your design uses chip resources in your planning. We are going to be imposing a CLB limit of 60 CLBs for check-point one. Yuh Meei's design used 45 CLBs and could have used fewer with more time.

Check-point one also offers your first chance at extra-credit on the project. There are two main areas where extra credit will be offered. The first is to make good use of the capability of the LCD to accommodate custom characters. It would be nice to have CD-player-related symbols displayed when buttons are pressed (along with the time, of course). The directions are in the OPTREX data sheet (and others). You will be allowed 75 CLBs total to do this. The second extra credit is easy, but could be really useful later in the project. It is possible to use a dip-switch to change the meaning of the keys (adding a bit to KEY[3:0]). Having the keyboard generate 32 unique outputs to the LCD would be extra credit opportunity number 2. You will be allowed 5 extra CLBs for this. Obviously, the two extra credits could also be done together as well (80 CLBs). Every point counts!

These extra credits must be finished before check off time to get credit.

Standard Character Font Table

High order bit / Low order bit	0000	0010	0011	0100	0101	0110	0111	1010	1011	1100	1101	1110	1111
X X X X0000	CG RAM (1)		0	@	P	`	p		—	タ	ミ	α	p
X X X X0001	(2)	!	1	A	Q	a	q	。	ア	チ	ム	ä	q
X X X X0010	(3)	"	2	B	R	b	r	「	イ	ツ	メ	ß	θ
X X X X0011	(4)	#	3	C	S	c	s	」	ウ	テ	モ	ε	∞
X X X X0100	(5)	$	4	D	T	d	t	、	エ	ト	ヤ	μ	Ω
X X X X0101	(6)	%	5	E	U	e	u	・	オ	ナ	ユ	σ	ü
X X X X0110	(7)	&	6	F	V	f	v	ヲ	カ	ニ	ヨ	ρ	Σ
X X X X0111	(8)	'	7	G	W	g	w	ァ	キ	ヌ	ラ	g	π
X X X X1000	(1)	(	8	H	X	h	x	ィ	ク	ネ	リ	√	x
X X X X1001	(2)	)	9	I	Y	i	y	ゥ	ケ	ノ	ル	-1	y
X X X X1010	(3)	*	:	J	Z	j	z	ェ	コ	ハ	レ	j	千
X X X X1011	(4)	+	;	K	[	k	{	ォ	サ	ヒ	ロ	x	万
X X X X1100	(5)	,	<	L	¥	l	|	ャ	シ	フ	ワ	¢	円
	(6)	-	=	M	]	m	}	ュ	ス	ヘ	ン	£	÷
	(7)	.	>	N	^	n	→	ョ	セ	ホ	゛	ñ	
	(8)	/	?	O	_	o	←	ッ	ソ	マ	°	ö	

Note: Character of high order bit 1110 and 1111 may be inadequate.

Name_____ Name_____

Project Checkpoint 1

Checkoff Sheet

Design

Master and sub-FSM state diagrams _____

Reduced Logic (K-Maps) _____

Implementation / Schematics

Clock Divider _____

Keypad Block _____

LCD Block _____

Testing

Trigger the logic analyzer of the commands to

Print "A" to the screen. _____

Everything Works _____

Finished 1st Week (full credit) _____

Finished 2nd Week (1/2 credit) _____

Extra Credit

1- Custom Symbols _____

2- KEY[4:0] _____