ESPECIFICAÇÃO DO PROJETO (Segunda Unidade) 2013.2

Projeto da CPU

O projeto da CPU precisa ser acoplado a um decodificador para display de sete segmentos que também será desenvolvido. Neste projeto da segunda unidade, a **ULA** e os registradores da CPU devem ser desenvolvidos usando Portas Lógicas ou em verilog, já a **Unidade de Controle** da CPU em Verilog.

O projeto deve estar pronto para baixar na placa de prototipação DE2-70, inclusive com as pinagens configuradas. As respectivas pinagens (associação de um sinal entrada/saída com um pino do FPGA) estão especificadas em anexo.

Deverá ser entregue um relatório impresso ao professor juntamente com o código fonte (pasta completa do projeto compactada), detalhando cada fase desenvolvida, que deve conter:

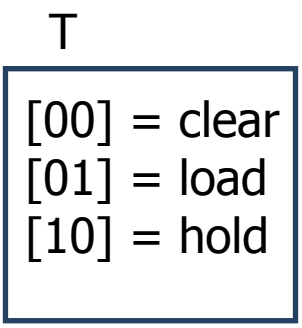
a) Capa com identificação dos alunos
b) Visão Geral do Projeto (figura ilustrando o sistema completo em blocos). Explicar sucintamente nesta etapa cada módulo desenvolvido.
c) Circuito ou Verilog projetado de cada módulo e simulação (waveform)
d) Circuito / Verilog com todo sistema conectado e simulação (waveform)
e) Conclusão

CPU

A CPU é composta da Unidade de Controle, Unidade Lógica e Aritmética e Registradores.

Registradores

Os registradores armazenam temporariamente um dado, que nunca estará no formato de complemento a dois. O registrador pode receber 3 comandos: (00, 01, 10), oriundos de Tx, Ty e Tz. Os registradores X, Y e Z possuem 6 bits cada um (um bit de sinal e cinco de módulo). A entrada para o registrador X está conectada à 5 switches, sendo um switch para o bit de sinal e os outros quatro para o módulo do registrador X. Percebe-se que tem um bit a mais internamente no registrador para tratar casos em que a soma de X + Y ultrapasse o máximo para 4 bits.



Unidade de Controle

A unidade de controle é responsável pela sequência de execução das instruções na CPU. O normal do sinal new_instruction é zero. Caso o "new_instruction = 1 e em seguida volte para zero", então indica que chegou uma nova instrução. Em seguida, o comando contido em "instruction" (oriundo do switch) é decodificado seguindo a Tabela1 abaixo. Exemplo: Caso "instruction=001", então foi o comando de CLEAR, com isso, deve-se colocar nos sinais Tx, Ty, Tz e Tula os valores indicados na tabela2. OBS.: Na tabela 2, os valores iguais a X na coluna correspondente a Tula significam don'tcare, ficando a cargo da equipe tratar este caso.

Tabela 1:

Instruction	**Ação**
001	CLEAR
010	ADD
011	SUB
100	DISP
101	LOAD_X

Tabela 2:

Ação	Tula	Tx	Ty	Tz
CLEAR	X	00	00	00
ADD	00	10	01	10
SUB	01	10	01	10
DISP	X	10	10	01
LOAD_X	X	01	10	10

Tula
[00]: ADD (+)
[01]: SUB (-)

ULA

A unidade lógica e aritmética é responsável pelas operações aritméticas de adição e subtração.

Entradas:

1. Dois vetores de 6 bits (1 para o sinal e 5 para o módulo) oriundos dos registradores X e Y, representando os operandos. Esses números são binários positivos ou negativos (não estarão complementados a 2).OBS.: O vetor do registrador X pode ter como valores números de -16 até 15, sendo limitado ao numero de switches. O vetor do registrador Y, por sua vez, possui números de -32 até 31.
2. O sinal Tula de 2 bits que representa o tipo de operação.

Saídas:

1. Um vetor F de 6 bits, que não deve estar complementado a dois, representando o resultado da operação.OBS.: O vetor F pode ter como valores números de -32 (100000) até 31 (011111).
2. Um sinal (LED) representando overflow (para as operações que podem gerar overflow).
3. Um sinal (LED) para indicar que o resultado é negativo (aceso quando negativo e apagado quando positivo).

Decodificador BCD para display de 7 segmentos

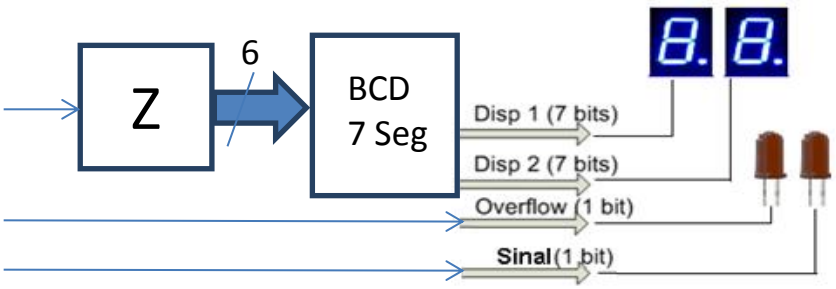
Para que seja possível exibir os números no display da placa DE2-70, o projeto da ULA deve estar acoplado a um decodificador. Os leds de cada display acendem quando colocado nível lógico 0.

Entradas:

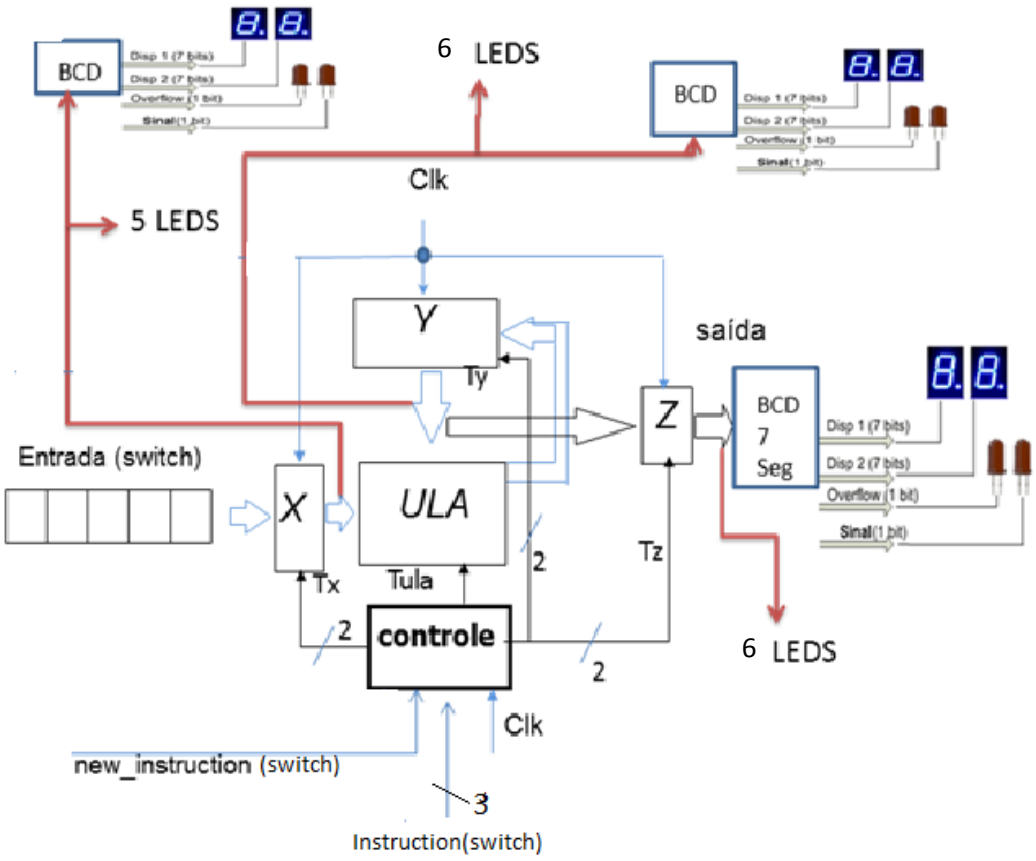
1. Vetor de 6 bits (F) vindo da ULA, representando um número binário positivo ou negativo (sem estar no complemento a 2).

Saídas:

1. Dois vetores de 7 bits representando os 2 displays, segundo a tabela abaixo.



Sistema Completo



Observação: Além de ser conectado a dois displays de 7 segmentos, cada registrador também deve estar conectado a 6 LEDS (cada bit ligado a um LED) vermelhos.

Observação: Os valores a serem processados virão de 5 switches, com 1 bit sendo de sinal e 4 para o módulo (ou seja, não virão complementados a dois).

Observação: Quando acontecer um overflow deverá ser mostrado no display de Z

Exemplo1:

SEQUÊNCIA: 7 + 2 - 3 - 1 = 5

Execução:

new_instruction (Switch) = e volta para zero

instruction (Switches): CLEAR (Zera Y, Z e X)

new_instruction (Switch) = e volta para zero

Entrada (Switches): 00111

instruction (Switches): LOAD_X [101] (carregaem X o valor 7)

new_instruction (Switch) = e volta para zero

instruction (Switches): ADD [010] (Adiciona Y = 0 com X = 7e carrega este valor em Y)

new_instruction (Switch) = e volta para zero

Entrada (Switches): 00010

instruction (Switches): LOAD_X [101] (carrega em X o valor 2)

new_instruction (Switch) = e volta para zero

instruction (Switches): ADD [010] (Adiciona Y = 7 com X = 2 e carrega este valor em Y)

new_instruction (Switch) = e volta para zero

Entrada (Switches): 00011

instruction (Switches): LOAD_X [101] (carrega em X o valor 3)

new_instruction (Switch) = e volta para zero

instruction (Switches): SUB [011] (Subtrai Y = 9 de X = 3 e carrega este valor em Y)

new_instruction (Switch) = e volta para zero

Entrada (Switches): 00001

instruction (Switches): LOAD_X [101] (carregaem X o valor 1)

new_instruction (Switch) = e volta para zero

instruction (Switches): SUB [011] (Subtrai Y = 6 de X = 1)

new_instruction (Switch) = e volta para zero

instruction (Switches): DISP [100] (Carrega o resultado em Z e mantém os valores em X e Y)

Exemplo2:

SEQUÊNCIA: 7 + 14-12 -5= 4

Execução:

new_instruction (Switch) = e volta para zero

instruction (Switches): CLEAR (Zera Y, Z e X)

new_instruction (Switch) = e volta para zero

Entrada (Switches): 00111

instruction (Switches): LOAD_X [101] (carregaem X o valor 7)

new_instruction (Switch) = e volta para zero

instruction (Switches): ADD [010] (Adiciona Y = 0 com X = 7 e carrega este valor em Y)

new_instruction (Switch) = e volta para zero

Entrada (Switches): 01110

instruction (Switches): LOAD_X [101] (carregaem X o valor 14)

new_instruction (Switch) = e volta para zero

instruction (Switches): ADD [010] (Adiciona Y = 7 com X = 14)

Overflow ocorre!

new_instruction (Switch) = e volta para zero

ready (habilita o instruction): 1

instruction (Switches): DISP [100] (Carrega em Z OU e zera X e Y)

Informações sobre clock

Signal Name	FPGA Pin No.	Description
CLK_28	PIN_E16	28 MHz clock input
CLK_50	PIN_AD15	50 MHz clock input
CLK_50_2	PIN_D16	50 MHz clock input
CLK_50_3	PIN_R28	50 MHz clock input
CLK_50_4	PIN_R3	50 MHz clock input
EXT_CLOCK	PIN_R29	External (SMA) clock input

Table 5.5. Pin assignments for the clock inputs.

Pinagem:

O FPGA Cyclone II para o qual o projeto será baixado é o EP2C70F896C6.

Signal Name	FPGA Pin No.	Description
LEDR[0]	PIN_AJ6	LED Red[0]
LEDR[1]	PIN_ AK5	LED Red[1]
LEDR[2]	PIN_AJ5	LED Red[2]
LEDR[3]	PIN_AJ4	LED Red[3]
LEDR[4]	PIN_AK3	LED Red[4]
LEDR[5]	PIN_AH4	LED Red[5]
LEDR[6]	PIN_AJ3	LED Red[6]
LEDR[7]	PIN_AJ2	LED Red[7]
LEDR[8]	PIN_AH3	LED Red[8]
LEDR[9]	PIN_AD14	LED Red[9]
LEDR[10]	PIN_AC13	LED Red[10]
LEDR[11]	PIN_AB13	LED Red[11]
LEDR[12]	PIN_AC12	LED Red[12]
LEDR[13]	PIN_AB12	LED Red[13]
LEDR[14]	PIN_AC11	LED Red[14]
LEDR[15]	PIN_AD9	LED Red[15]
LEDR[16]	PIN_AD8	LED Red[16]
LEDR[17]	PIN_AJ7	LED Red[17]

Signal Name	FPGA Pin No.	Description
KEY[0]	PIN_T29	Pushbutton[0]
KEY[1]	PIN_T28	Pushbutton[1]
KEY[2]	PIN_U30	Pushbutton[2]
KEY[3]	PIN_U29	Pushbutton[3]

Table 5.2. Pin assignments for the pushbutton switches.

Signal Name	FPGA Pin No.	Description
LEDG[0]	PIN_W27	LED Green[0]
LEDG[1]	PIN_ W25	LED Green[1]
LEDG[2]	PIN_ W23	LED Green[2]
LEDG[3]	PIN_ Y27	LED Green[3]
LEDG[4]	PIN_ Y24	LED Green[4]
LEDG[5]	PIN_ Y23	LED Green[5]
LEDG[6]	PIN_ AA27	LED Green[6]
LEDG[7]	PIN_ AA24	LED Green[7]
LEDG[8]	PIN_ AC14	LED Green[8]

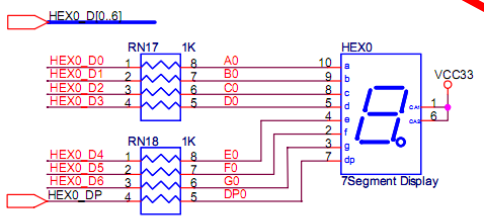
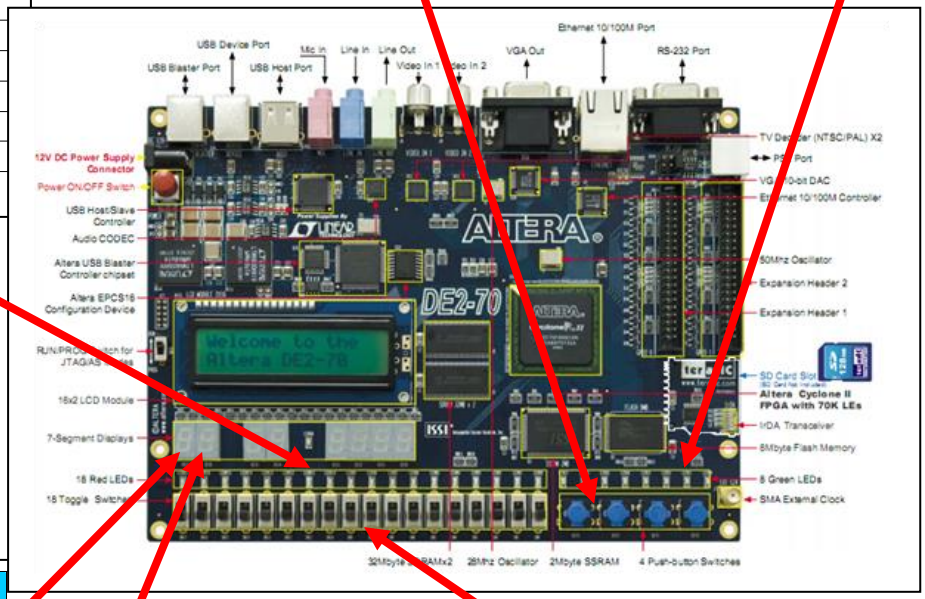


Figure 5.6. Schematic diagram of the 7-segment displays.



Signal Name	FPGA Pin No.	Description
HEX0_D[0]	PIN_AE8	Seven Segment Digit 0[0]
HEX0_D[1]	PIN_AF9	Seven Segment Digit 0[1]
HEX0_D[2]	PIN_AH9	Seven Segment Digit 0[2]
HEX0_D[3]	PIN_AD10	Seven Segment Digit 0[3]
HEX0_D[4]	PIN_AF10	Seven Segment Digit 0[4]
HEX0_D[5]	PIN_AD11	Seven Segment Digit 0[5]
HEX0_D[6]	PIN_AD12	Seven Segment Digit 0[6]
HEX0_DP	PIN_AF12	Seven Segment Decimal Point 0

Signal Name	FPGA Pin No.	Description
HEX1_D[0]	PIN_ AG13	Seven Segment Digit 1[0]
HEX1_D[1]	PIN_ AE16	Seven Segment Digit 1[1]
HEX1_D[2]	PIN_ AF16	Seven Segment Digit 1[2]
HEX1_D[3]	PIN_AG16	Seven Segment Digit 1[3]
HEX1_D[4]	PIN_AE17	Seven Segment Digit 1[4]
HEX1_D[5]	PIN_AF17	Seven Segment Digit 1[5]
HEX1_D[6]	PIN_AD17	Seven Segment Digit 1[6]
HEX1_DP	PIN_ AC17	Seven Segment Decimal Point 1

Signal Name	FPGA Pin No.	Description
SW[0]	PIN_AA23	Toggle Switch[0]
SW[1]	PIN_AB26	Toggle Switch[1]
SW[2]	PIN_AB25	Toggle Switch[2]
SW[3]	PIN_AC27	Toggle Switch[3]
SW[4]	PIN_AC26	Toggle Switch[4]
SW[5]	PIN_AC24	Toggle Switch[5]
SW[6]	PIN_AC23	Toggle Switch[6]
SW[7]	PIN_AD25	Toggle Switch[7]
SW[8]	PIN_AD24	Toggle Switch[8]
SW[9]	PIN_AE27	Toggle Switch[9]
SW[10]	PIN_W5	Toggle Switch[10]
SW[11]	PIN_V10	Toggle Switch[11]
SW[12]	PIN_U9	Toggle Switch[12]
SW[13]	PIN_T9	Toggle Switch[13]
SW[14]	PIN_L5	Toggle Switch[14]
SW[15]	PIN_L4	Toggle Switch[15]
SW[16]	PIN_L7	Toggle Switch[16]
SW[17]	PIN_L8	Toggle Switch[17]

Table 5.1. Pin assignments for the toggle switches.