PLAGIARISM DECLARATION

1. I know that plagiarism means taking and using the ideas, writings, works or inventions of another as if they were one's own. I know that plagiarism not only includes verbatim copy- ing, but also the extensive use of another person's ideas without proper acknowledgement (which includes the proper use of quotation marks). I know that plagiarism covers this sort of use of material found in textual sources and from the Internet.
2. I acknowledge and understand that plagiarism is wrong
3. I understand that my research must be accurately referenced. I have followed the rules and conventions concerning referencing, citation and the conventions concerning referencing and citation.
4. This assignment is my own work, or my group's own unique group assignment. I acknowledge that copying someone else's assignment, or part of it, is wrong, and that submitting identical work to others constitutes a form of plagiarism.
5. I have not allowed, nor will I in the future allow, anyone to copy my work with the intention of passing it off as their own work.

Group Name	Group 2

Name	Titus Tong Zhong Xu
Matric Number	A21EE0298
Signature	[illegible]
Date	4/7/2023

Name	Chieng You Wei
Matric Number	A21EE0035
Signature	[illegible]
Date	4/7/2023

Name	Sim Zhong Xian
Matric Number	A21EE0194
Signature	[illegible]
Date	4/7/2023

MARKING RUBRIC: M5

Group Name	Group 2	Section	9 & 10
Team Member 1	Titus Tong Zhong Xu		
Team Member 2	Chieng You Wei		
Team Member 3	Sim Zhong Xian		

Item	Category	0	1	2	3	Score
1.	**Top-Down Design:** Approaches problem at high-level, then breaks the problem into smaller, more manageable pieces	Fails to grasp the concept of top-down design.	Understands the concept of top-down design.	Somewhat understands the concept of top-down design.	Fully understands and implements the full top-down design procedure and abstraction.	
2.	**Bottom-Up Implementation:** Start with the lowest level, ends with the top-level circuit	Seemingly random collections of circuits.	Mostly correct sequence.	Mostly incorrect sequence.	Bottom-up sequence is clearly evident.	
3.	**Documentation / Schematics:** Complete & clear. Proper signal labels.	Missing diagrams.	Hard to read diagrams. Improper use of labels.	Mostly complete.	Complete & clear documentation.	
4.	**Simulation Analysis:** Brief & direct to the point	Missing analysis.	Unsure what is important.	Reasonably easy to read, with minimum errors	Clear & easy to read reporting. Important results are highlighted.	
5.	**Test Data:** Covers all important test cases with no redundant cases	Missing simulation or no hardware.	Too few test cases and/or too many repeated test cases.	Covers most test cases and/or some repeated inputs.	Proper choice of test inputs.	
6.	**Hardware Functionality:** Working circuit	No hardware or hardware does not work.	Works but not to specifications.	Works according to specifications, sometimes.	Works flawlessly.	×4
7.	**Hardware Presentation:** Easy on the eyes.	No hardware.	A jumble of wires.	Acceptably neat.	Very neat.	
					Raw Marks	
					Final Assignment Marks $(\frac{Raw}{30} \times 5)$	

THE PLAGIARISM DECLARATION MUST BE SIGNED BY ALL TEAM MEMBERS AND THERE MUST NOT BE ANY FORM OF PLAGIARISM TO RECEIVE ANY MARKS.

Milestone 5: Datapath Unit (DU) Design

Step 1

Determine all of the required blocks (i/o, function, size) to implement the DU. Design and verify each block in the DU using QuartusII. Reuse the blocks which have been designed in the previous milestones. Apply hierarchical design concept in your design. Name the blocks according to its function. Highlight/annotate important information in the output waveform.

Print the following pages as PDF in landscape format for all of the blocks. Use page numbering as below accordingly. -1 refers to block 1, -2 refers to block 2 etc.
Page 1-1: Circuit schematic
Page 2-1: Compilation report
Page 3-1: Annotated simulation output waveform.

Step 2

Connect all of the blocks as determined in Step 1 and name it as DU in Quartus. Verify the operation of the datapath unit by providing the correct combination and sequence of inputs. Highlight/annotate important information in the output waveform.

Print the following pages as PDF in landscape format.
Page 4: Circuit schematic
Page 5: Compilation report
Page 6: Annotated simulation output waveform.

Step 3

Create symbol for the DU.

Step 4

Combine all of the PDF pages into 1 PDF document. Add the plagiarism declaration as the front cover. Each team member must sign the declaration. Upload to elearning.utm.my.

Option 8 : 8-bit Binary to BCD Converter

Implement 8-bit Binary to BCD Converter based on algorithm and datapath unit shown in Figure 8-1 and Figure 8-2.

Algorithm and Block Diagram of DU

```
// Double-dabble algorithm
Hundreds = 0;
Tens = 0;
Ones = 0;
for (i=0; i<8; i++ {
  // check all columns >= 5
  if (Hundreds >= 5) Hundreds += 3;
  if (Tens >= 5) Tens += 3;
  if (Ones >= 5) Ones += 3;
  // shift all bits left
  Hundreds <<= 1;
  Hundreds[0] = Tens[3];
  Tens <<= 1;
  Tens[0] = Ones[3];
  Ones <<= 1;
  Ones[0] = Binary[7]
  Binary <<= 1;
}
```

Converting 255 from binary to BCD:

Hundreds	Tens	Ones	Binary	Oper.
			1111 1111	Load
		1	111 1111	<< #1
		11	11 1111	<< #2
		111	1 1111	<< #3
		1010		+3
	1	0101	1111	<< #4
		1000		+3
	11	0001	111	<< #5
	110	0011	11	<< #6
	1001			+3
1	0010	0111	1	<< #7
		1010		+3
10	0101	0101		<< #8

Figure 6-1: *Algorithm: Pseudocode and iteration example.*

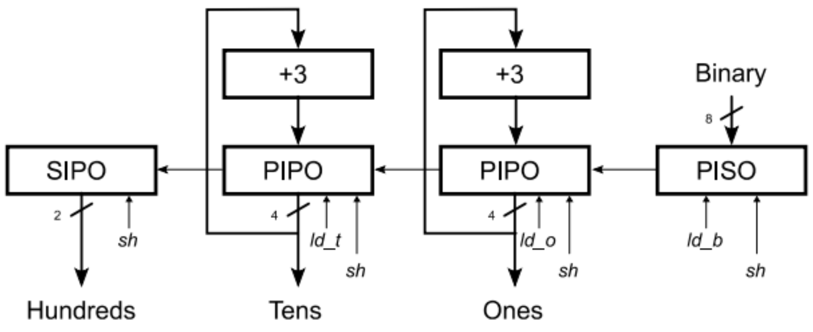


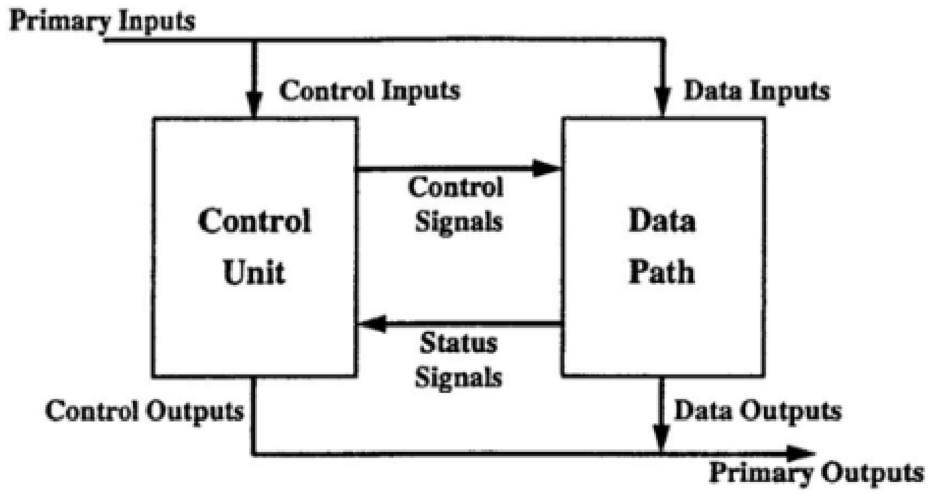
Figure 8-2: *Binary to BCD Datapath unit.*

Datapath Components

Component	Details
Conditional add 3 module	4-bit
PISO Shift left Register	8-bit with Load and Enable
PIPO Shift left Register	4-bit with Load and Enable
SIPO Shift left Register	2-bit with Enable

5.1 Introduction

Structurally, the data processor is built up of datapath unit and control unit as shown in Figure below. In this case, a datapath unit (DU) of the 8-bit binary-to-BCD converter design which is assigned for converting 8-bit binary number to BCD number. DU executes arithmetic and logic functions to solve an algorithm. 8-bit binary-to-BCD Converter is a component which reads in 8 bit binary number from user logic over a parallel interface and outputs the binary coded decimal (BCD) equivalent based on the idea of double-dabble algorithm. This algorithm is using the basic idea of shift register and conditional adders (Cadd3). In the DU of 8-bit binary-to-BCD converter, there are mainly registers (shift register and data register) and conditional adders. In order to produce a status signal, a level-to-pulse converter is used since its characteristic could fulfill the needs.



5.2 Identify and Construct the Building Blocks

5.2.1 Conditional add 3 modules (4-bit) taken from Milestone 3 with some modification

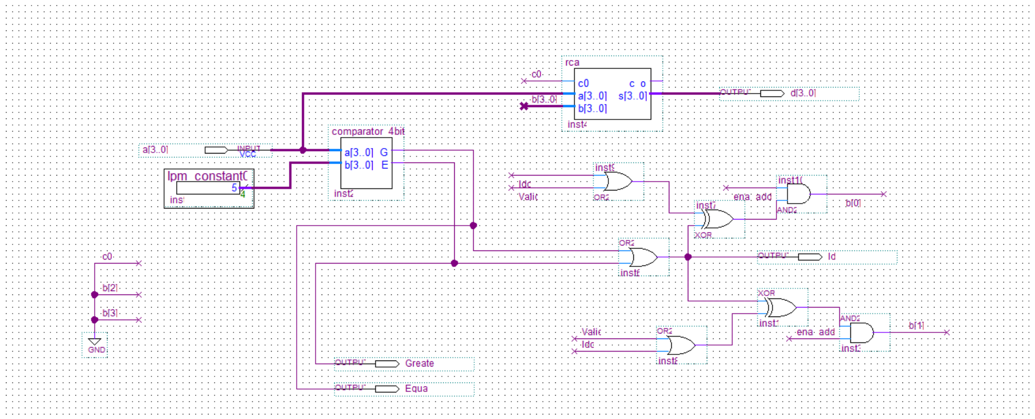


Figure 1: Cadd3 module

We have made modifications to the CADD3 module from Milestone 3 by integrating it with a 4-bit comparator. This comparator outputs a HIGH signal when the input a[3..0] from PIPO is greater than or equal to 5. To achieve this, we set the data inputs for b[3..0] at the comparator to a constant value of 0101 (5 in decimal).

For the RCA module, the data input for b[3..0] is generated based on certain conditions. When the add3 operation is required, the data input is set to 0011 (3 in decimal), and when the 8-bit data from PISO has finished shifting, the data input is set to 0000.

Additionally, we have implemented a simple combinational circuit consisting of an OR gate, an XOR gate, and an AND gate. This circuit generates the values for b[0] and b[1], which are inputs for the RCA module. The OR gate receives the load counter (ldc) signal as well as the valid signal, ensuring that its output is always LOW while the shifting is not done. The XOR gate produces a HIGH output when the GE signal is high and the ena_add signal from the control unit (CU) is also high, allowing the add3 operation to occur. In the case where the 8-bit shifting is complete and ena_add from the CU is LOW, the add3 operation is blocked, and the value is held. The inputs b[3..2] and c0 for the RCA module are always set to 0. Furthermore, there is an OR gate connected to the output pin, which serves as a control signal for the ld_o and ld_t signals of the multiplexer. This allows the output of the RCA module to be loaded into the flip-flop.

PISO Shift Left Register (8 bits with load and enable)

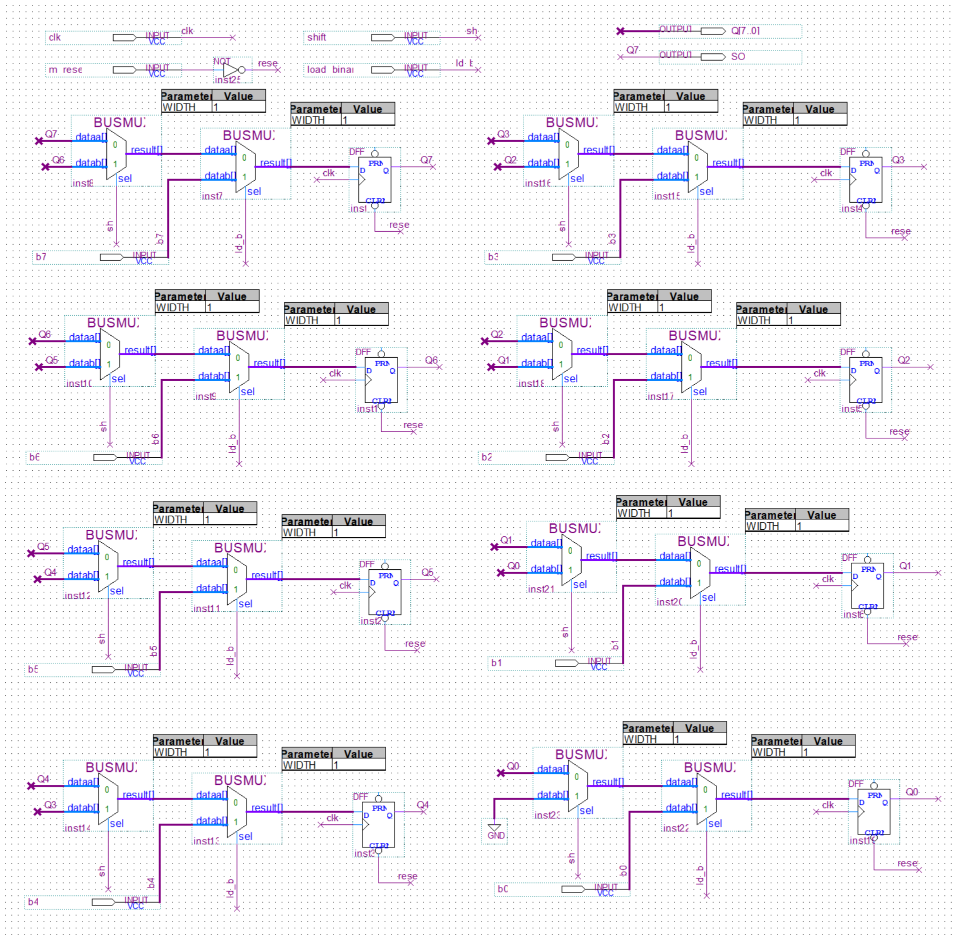


Figure 2: PISO shift left register.

The Parallel Input Serial Output (PISO) module is the initial module in the shifter. It consists of two 2:1 multiplexers and one D flip flop for each bit of input data b[7..0]. The primary control signal for the multiplexers is load binary (ld_b). When ld_b is 1, the input data is taken from the DIP switch. When ld_b is 0, the output from the secondary multiplexer is selected. The secondary control signal for the multiplexers is shift (sh). When sh is 1, the data is shifted left by one bit, and when sh is 0, the data is held.

The Most Significant Bit (MSB) is connected to the output pin SO, allowing it to be shifted to the next module in the shifter. Since we have 8 bits, this concept is duplicated seven more times for the remaining bits. It is important to note that the secondary multiplexer for Q0 has a specific behavior. When sh is 1, it will output a value of 0 since pin 1 is connected to GND. Therefore, when all eight bits are shifted out, the output will be 0000 0000, indicating that the shifting process is complete.

Take the first PISO left shift register cell (Left top) as an example: when Shift = 0 and ld_b = 0, $Q7^+ = Q7$; when Shift = 0 and ld_b = 1, $Q7^+ = b7$; when Shift = 1 and ld_b = 0, $Q7^+ = Q6$; when Shift = 1 and ld_b = 1, $Q7^+ = b7$. The operation of the register cell is summarised below:

ld_b	Sh	$Q7^+$	Function
0	0	Q7	Hold
0	1	Q6	Shift left
1	x	b7	Load

Table 1: Truth table for selectors of 2to1 mux, ld_b and sh and their functions.

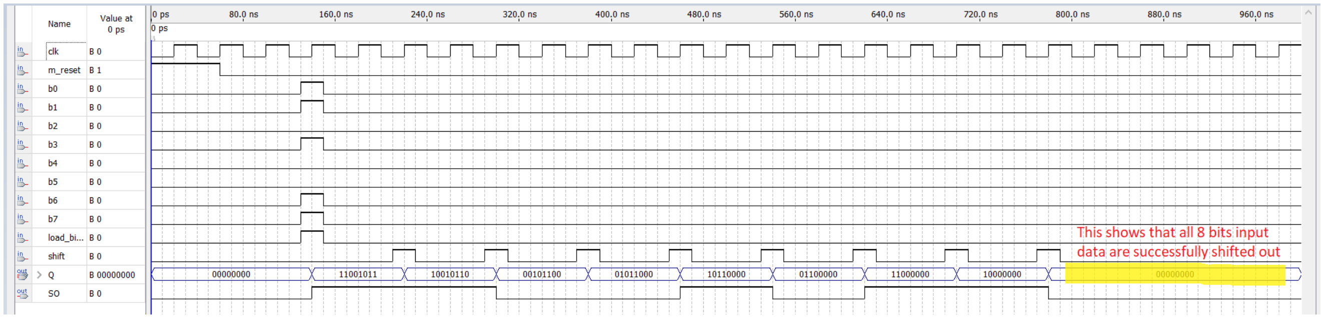
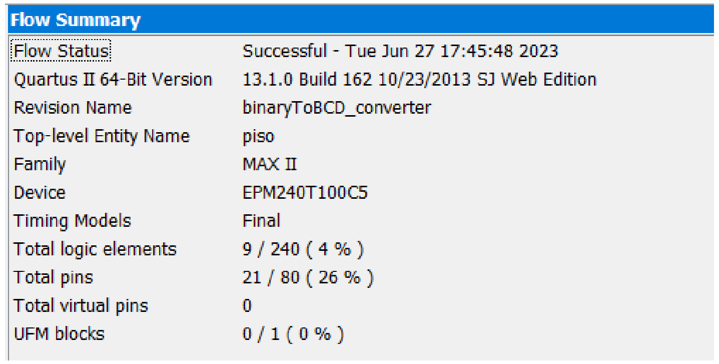


Figure 3: Waveform of PISO Shift Left Register



Flow Summary

Flow Status	Successful - Tue Jun 27 17:45:48 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	piso
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	9 / 240 (4 %)
Total pins	21 / 80 (26 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

Figure 4: Compilation Report of PISO Shift Left Register

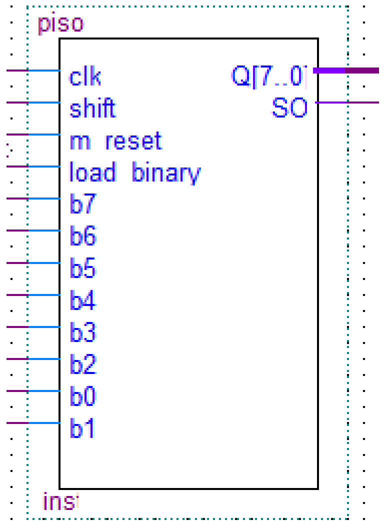


Figure 5: Symbol Diagram of PISO Shift Left Register

First PIPO Shift Left Register (4 bits with load and enable)

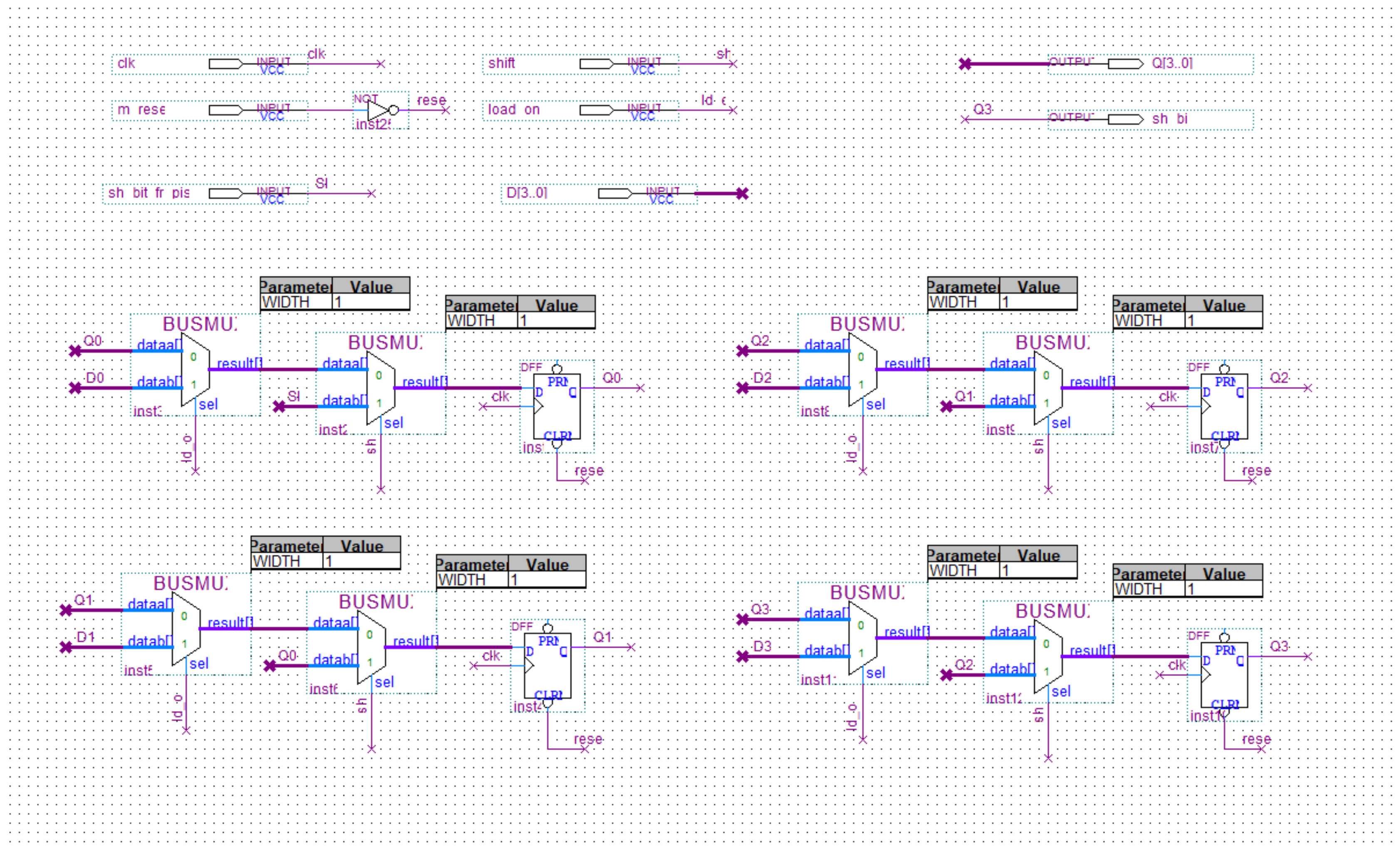


Figure 6: Schematic Diagram of first PIPO Shift Left Register (4 bits with load and enable)

Similarly, 4 bits PIPO shift left register consists of two 2 to 1 busmux (shift and ld_o) and 1 D flip-flop for each cell. D[3..0] is the data from the cadd3 module, sh and ld_o are controlled by CU. By taking the first PIPO register cell (Left top), when Shift = 0 and ld_o = 0, $Q0^+ = Q0$; when Shift = 0 and ld_b = 1, $Q0^+ = D0$; when Shift = 1 and ld_o = 0, $Q0^+ =$ SI (serial input from PISO); when Shift = 1 and ld_o = 1, $Q0^+ =$ SI. The operation of the register cell is summarised below:

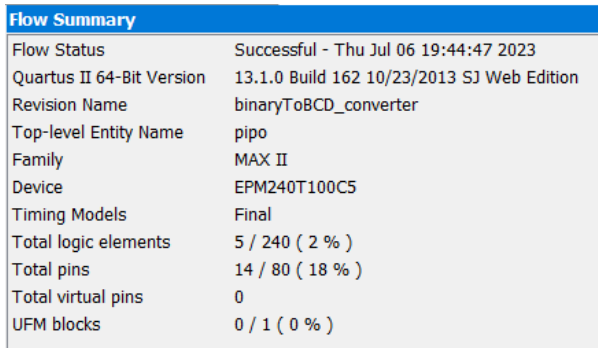
Sh	ld_o	$Q0^+$	Function
0	0	Q0	Hold
0	1	D0	Input data from cadd3
1	x	SI	Shift in input from PISO

Table 2: Truth table for first PIPO register cell

Sh	ld_o	$Q1^+$	Function
0	0	Q1	Hold
0	1	D1	Input data from cadd3
1	x	Q0	Shift left

Table 3: Truth table for second PIPO register cell

The 2 other register cells are modified based on Table 2 and 3.



Flow Summary	
Flow Status	Successful - Thu Jul 06 19:44:47 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	pipo
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	5 / 240 (2 %)
Total pins	14 / 80 (18 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

Figure 7: Compilation Report of first PIPO Shift Left Register (4 bits with load and enable)

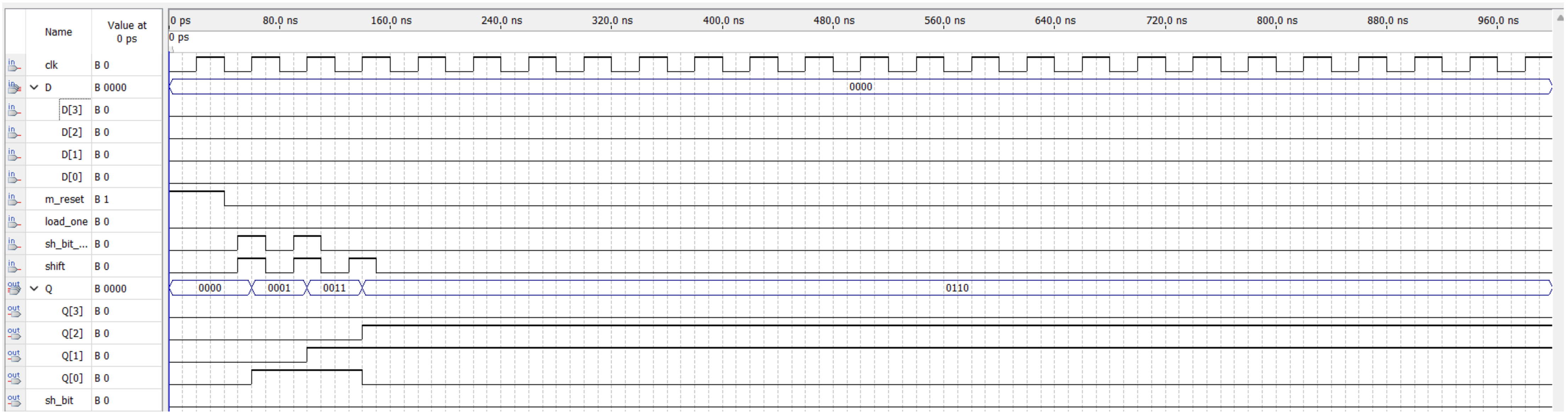


Figure 8: Generated waveform of PIPO Shift Left Register

When the shift bit given is 1 and the shift signal is given the original number 0000 will shift and become 0001. The number 0001 becomes 0011 since there is a high for both shift bit and shift signal. And if the signal shift bit is low, while the shift signal is high, this shift 0 into the previous number, and it now becomes 0110.

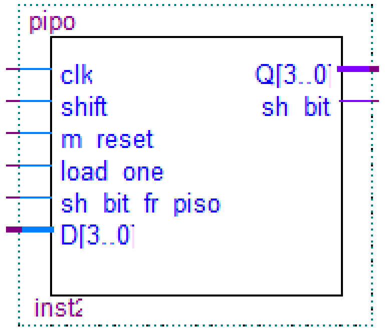


Figure 9: Symbol Diagram of first PIPO Shift Left Register (4 bits with load and enable)

Second PIPO Shift Left Register (4 bits with load and enable)

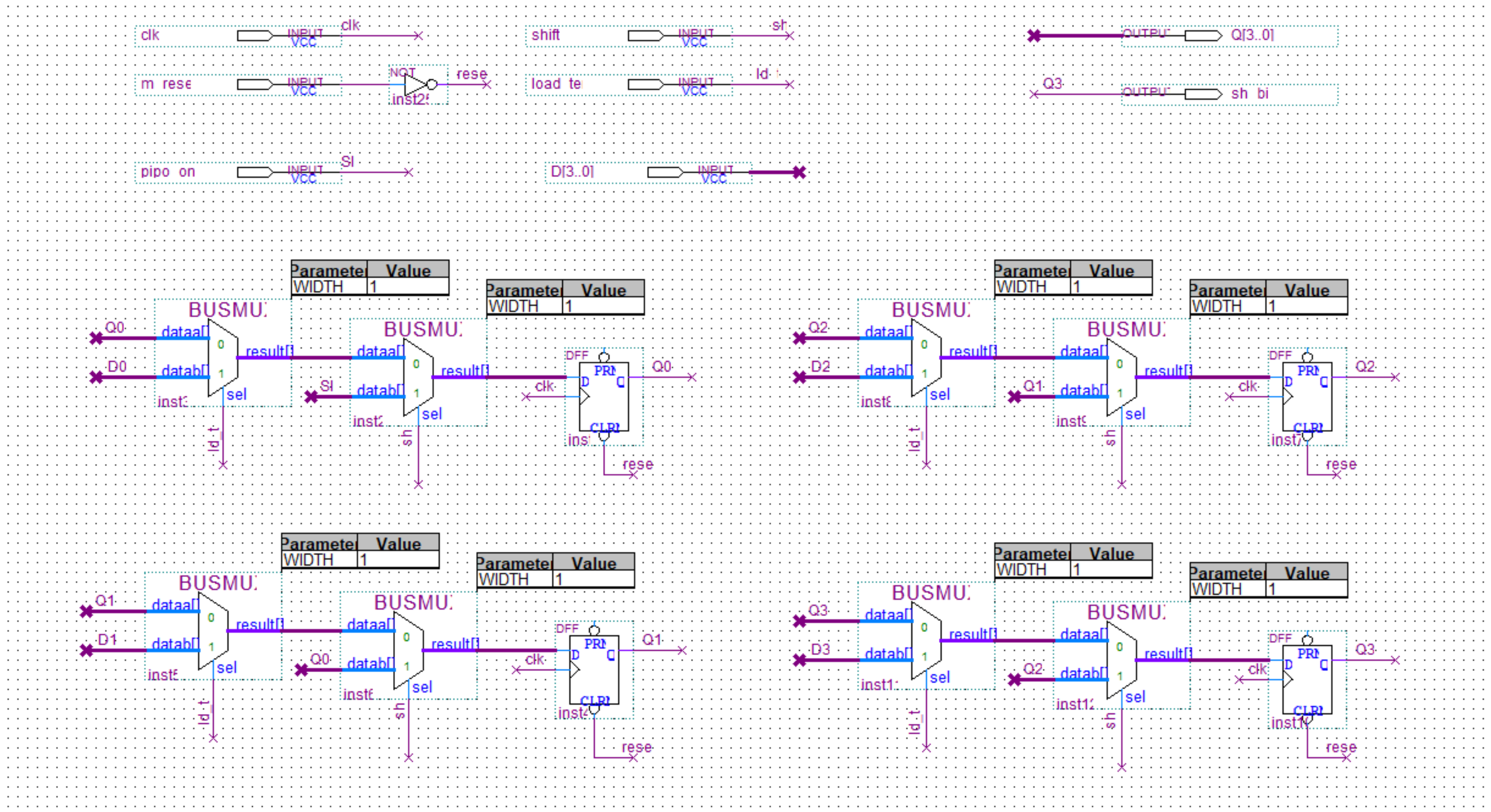
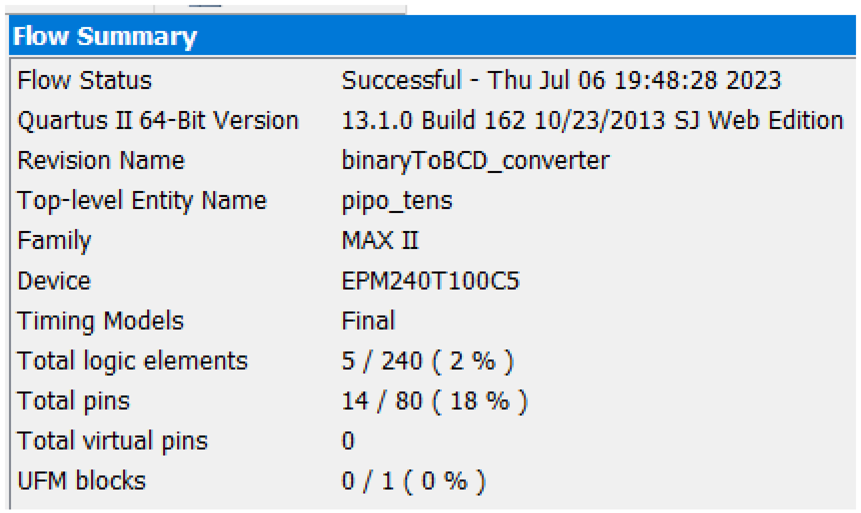


Figure 10: Schematic Diagram of second PIPO Shift Left Register (4 bits with load and enable)

Since its schematic is totally the same except the selector ld_o change to ld_t, and the waveform is similar to the first PIPO Shift Left Register, refer to Figure 6.



Flow Summary

Flow Status	Successful - Thu Jul 06 19:48:28 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	pipo_tens
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	5 / 240 (2 %)
Total pins	14 / 80 (18 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

Figure 11: Compilation Report of first PIPO Shift Left Register (4 bits with load and enable)

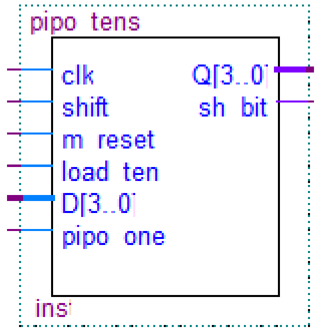
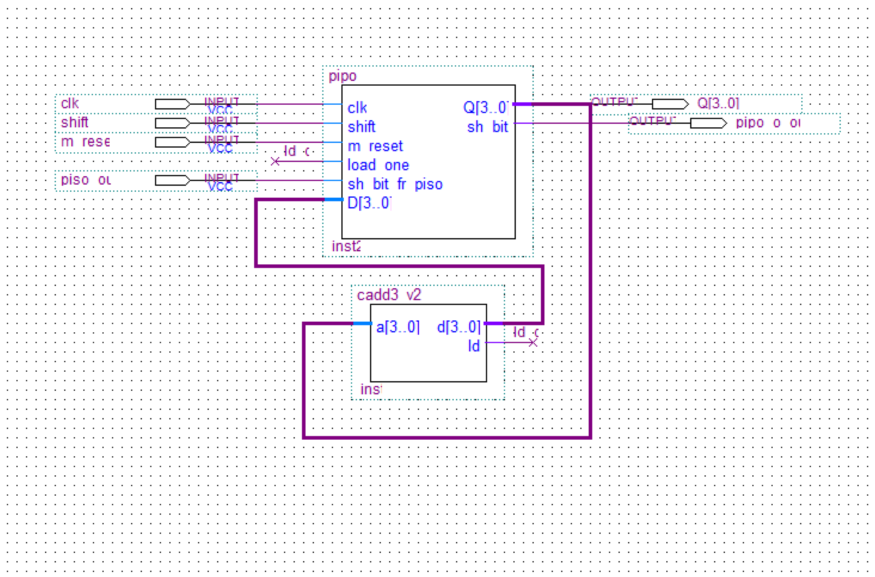


Figure 12: Symbol Diagram of first PIPO_Tens Shift Left Register (4 bits with load and enable)

PIPO Shift Left Register (4 bits with load and enable) with cadd3



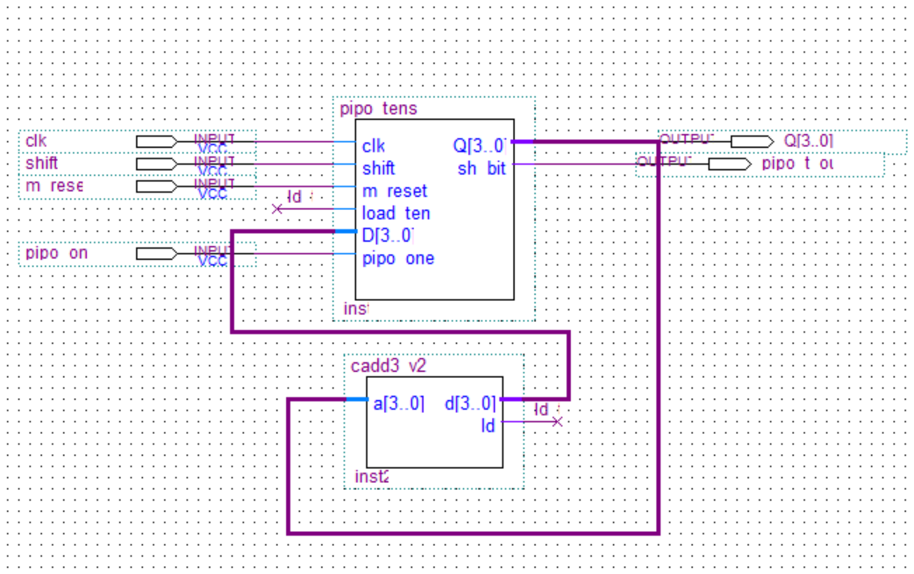
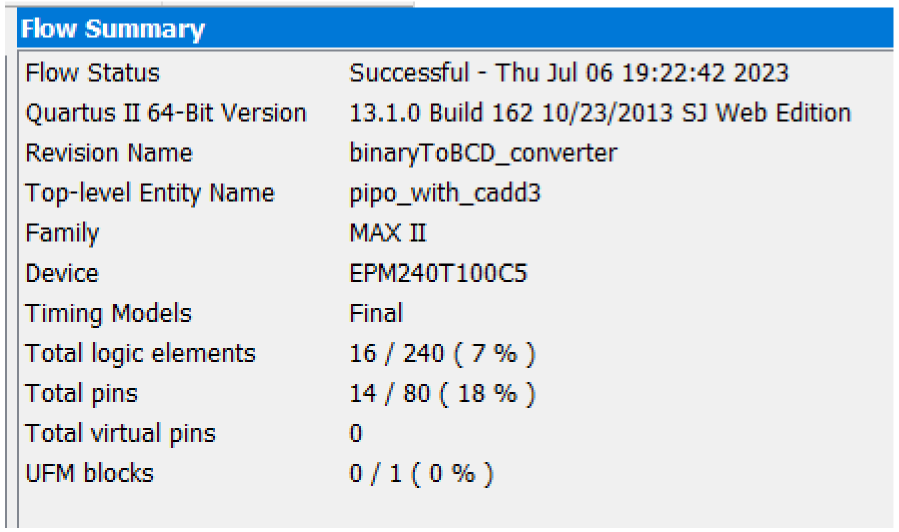


Figure 14: PIPO Shift Left register (Ones)

Figure 13: PIPO Shift Left register (Tens)

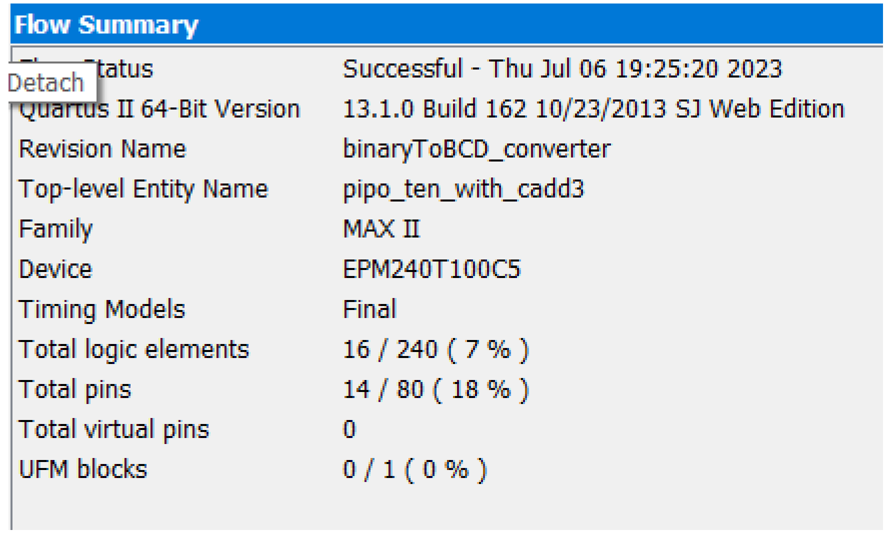
Figure 8 and 9 show the connection for cadd3 module and PIPO register. The input of the cadd3 module is from PIPO output Q[3..0] and the output of cadd3 goes into PIPO.



Flow Summary

Flow Status	Successful - Thu Jul 06 19:22:42 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	pipo_with_cadd3
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	16 / 240 (7 %)
Total pins	14 / 80 (18 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

Figure 15 : Compilation Report of PIPO Shift Left register (Ones)



Flow Summary

Flow Status	Successful - Thu Jul 06 19:25:20 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	pipo_ten_with_cadd3
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	16 / 240 (7 %)
Total pins	14 / 80 (18 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

Figure 16 : Compilation Report of PIPO Shift Left register (Tens)

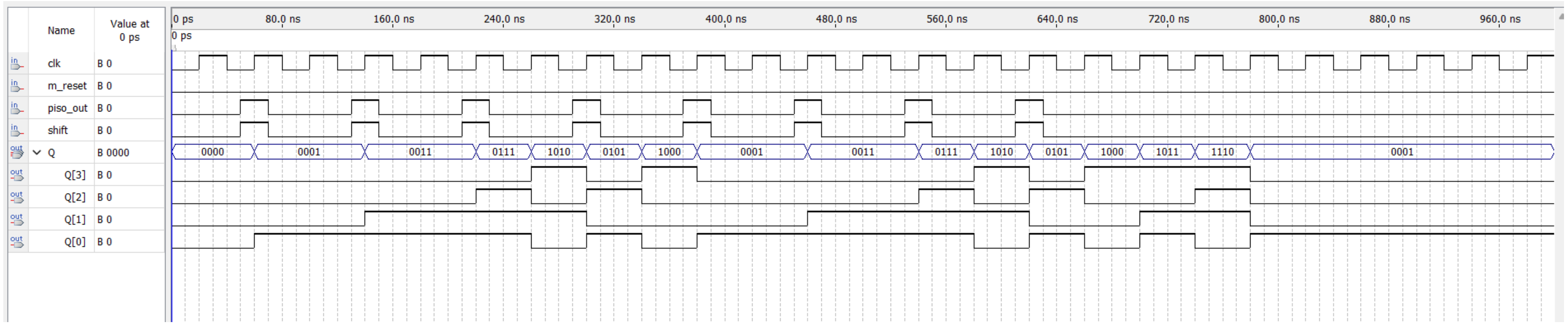


Figure 17: Waveform of PIPO and cadd3 module

The output from PIPO will be compared by the cadd3 module with a comparator. If it is larger than 5, cadd3 will add 3 to the original value before shifting happens.

SIPO Shift Left Register (2 bits with enable)

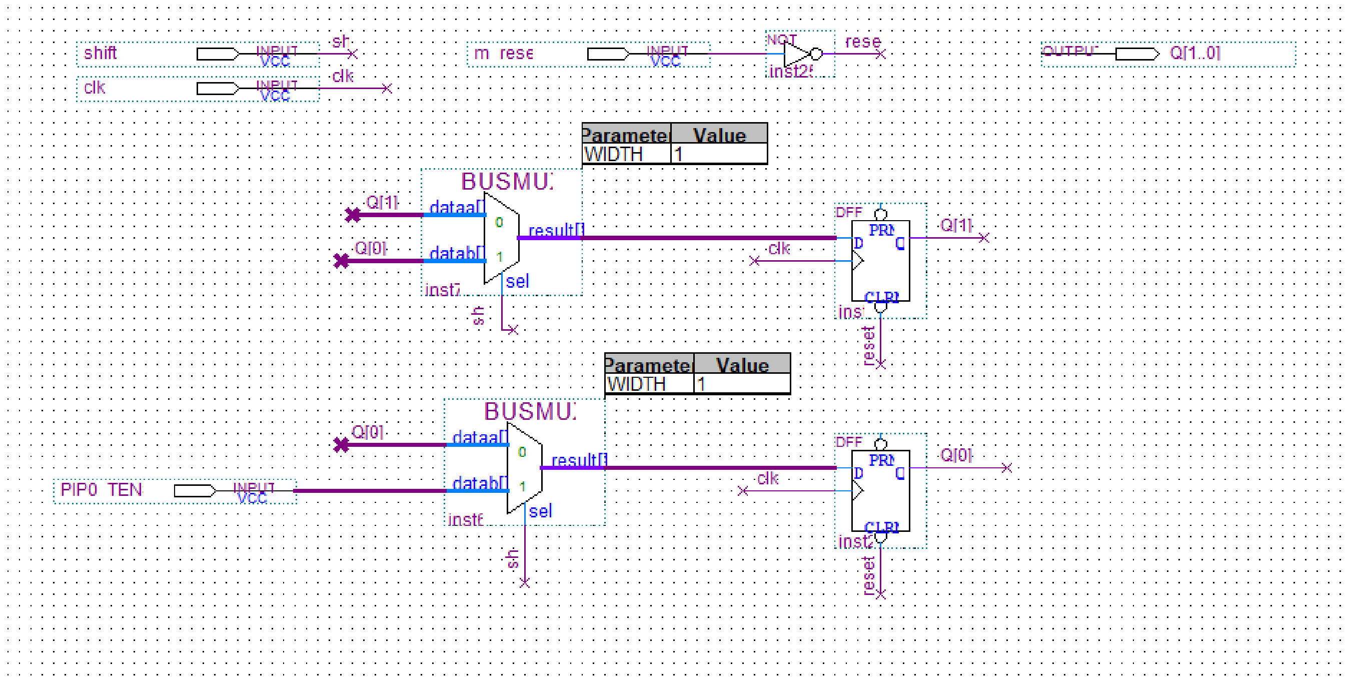
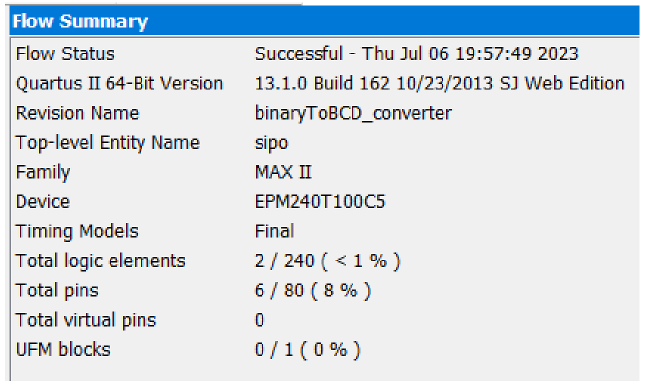


Figure 18: Schematic diagram for SIPO shift left register

The last register in this binary to BCD is the SIPO shift left register. This 2 bits register carries the bits for hundreds of the BCD.



Flow Summary	
Flow Status	Successful - Thu Jul 06 19:57:49 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	sipo
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	2 / 240 (< 1 %)
Total pins	6 / 80 (8 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

Figure 19: Compilation Report for SIPO shift left register

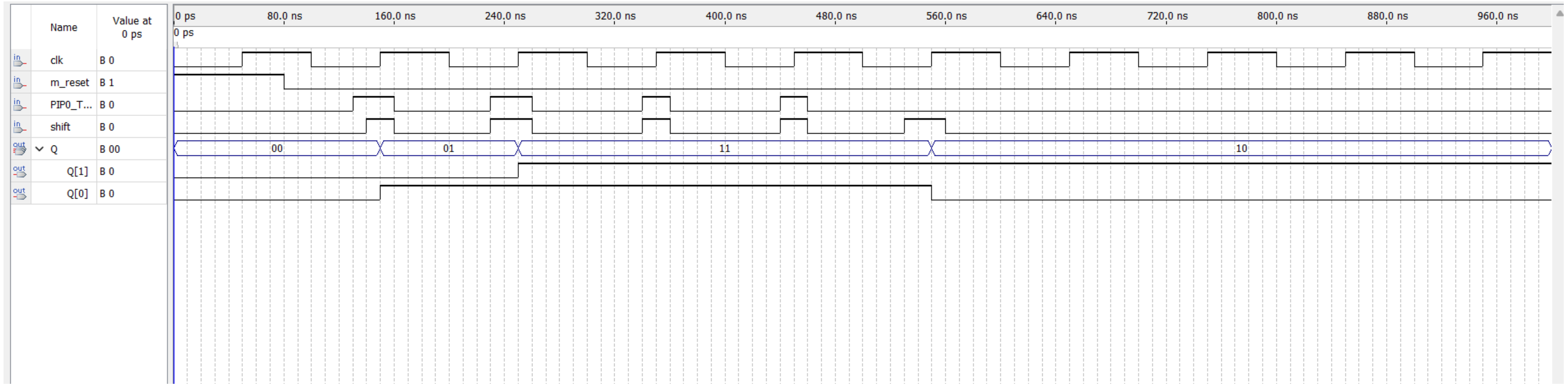


Figure 20: Waveform of SIPO shift left register

Similar to the PIPO register, when the PIPO output is 1 and shift is 1, binary number 00 shifts 1 bit left to become 01. The function is summarised in the table below:

Output from PIPO (tens)	Shift	Original Number (10)	Function
0	0	10	No shift (Hold)
0	1	00	Shift 0 into 10
1	0	10	No shift (Hold)
1	1	01	Shift 1 into 10

Table 4: Function of SIPO Shift left register.

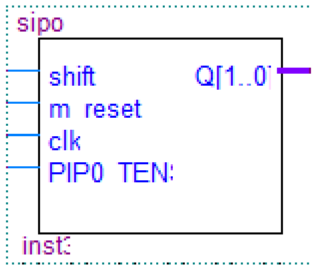


Figure 21: Symbol diagram for SIPO shift left register

Integration of Block Diagrams (DU)

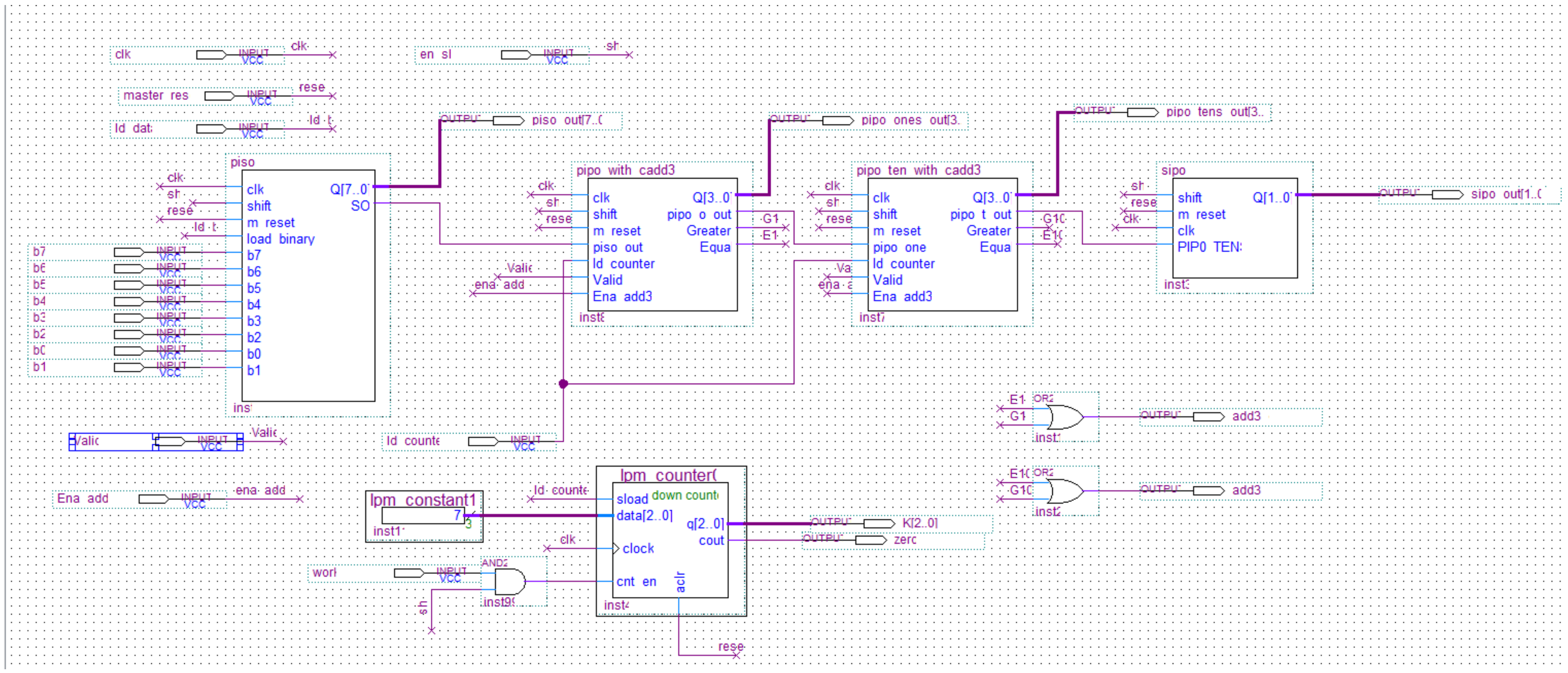
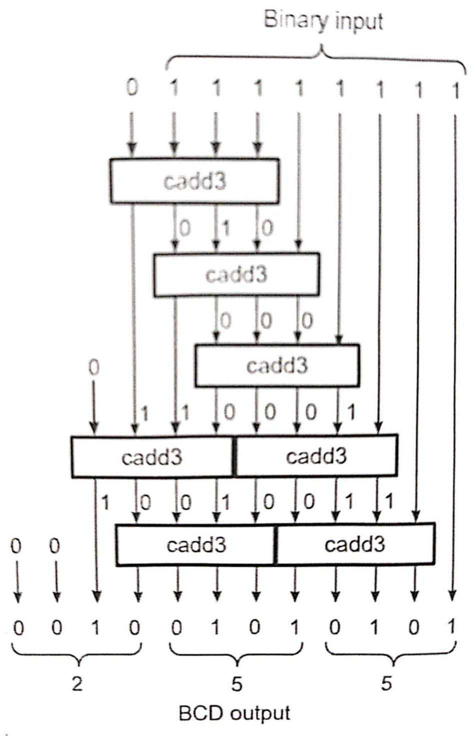


Figure 22: Schematic Diagram of DU

The title assigned is 8 bit Binary-to-BCD Converter which means the binary input is parallel in and is converted to BCD numbers.numbers. Since each bit has two possible binary values which are 0 or 1, we designed them by using 8 bit DIP(Dual In-Line Package) switches. It is a small switch in a DIP that is packaged with others in a group to hold configurations and select the interrupt request. It also is a set of manual electrical switches. So we need to set the 8 bit binary number which we want to convert on the 8 bit DIP switch so that the input will be parallel into the circuit. After the 8 bit binary number was set on the DIP switch, we designed a switch which allows the 8-bit binary number to enter the DU circuit. In DU here, it is set as ld_data, which will load the 8 bit binary into the DU and it is controlled by the CU circuit. Besides ld_data, we also designed some inputs were controlled by CU circuits which are master_reset, en_shift, en_add, work and valid. Master_reset is going to reset all values, including 8-bits binary load data. We designed it as an active high. Once the master_reset button is being pressed, all data will become zero and wait for the new data in. En_shift was designed to enable the shifting process in each designed block. When it is input one, the 8-bit binary will start shifting from PISO register to PIPO_Ones register, then PIPO_Ones register to PIPO_Tenths register, then PIPO_Tenths register to SIPO register. En_add was designed to enable the adding of 3 in the PIPO_Ones register and PIPO_Tenths register. When the 4-bit binary received in the PIPO_Ones register or PIPO_Tenths register is greater than 5 or equal to five, the pulse(logic 1) will be stimulated from the CU to enable adding of 3(0011) to the current 4 bit binary. At the same time, the en_shift will be logic 0 in order to stop shifting. Two processes cannot happen at the sametimes. However, when it managed to finish one adding process. The next process is to perform shifting regardless if the 4 bit binary in the register is greater than or equal to five. Once all the 8 bit binary is shifted out from the PISO register. It will stop the process of shifting, the valid become 1 and it is generated by CU.

In general, in PISO register, we load the 8-bit data in parallel, then the output of this data will be in serial. Next, in PIPO_Ones register and PIPO_Tenths register, they are 4-bit registers and are performing the same function which is received the data from PISO register in parallel and continue to do shifting from PIPO_Ones register to PIPO_Tenths register. When 4-bit binary number exceed 3, it will perform the adding process in the registers. Then the last register is SIPO register, it is a 2-bit register, so it will no greater than or equal to 5. It received the data from PIPO_Tenths register in serial because it is SIPO (Serial In Parallel Output) register then parallel out. So in our designed DU, it can process 8 bit binary number so meaning it can display the BCD from 0 to 255(the greatest number).

On the other hand, in the Figure 22, obviously we can see that we have design a Imp counter. It is used to perform counting when one bit is successfully shifting from the PISO register to PIPO_Ones register. So, work is designed at here, when one bit is completely shifted from PISO register to PIPO_Ones register. CU will sent the signal to work and it will work the counter. It is designed as 3 bit based, so it is just counted from 7 to 0.



(b) Eight-bit binary input.

Figure 23: Flow of DU when 11111111 as 8-bit binary input

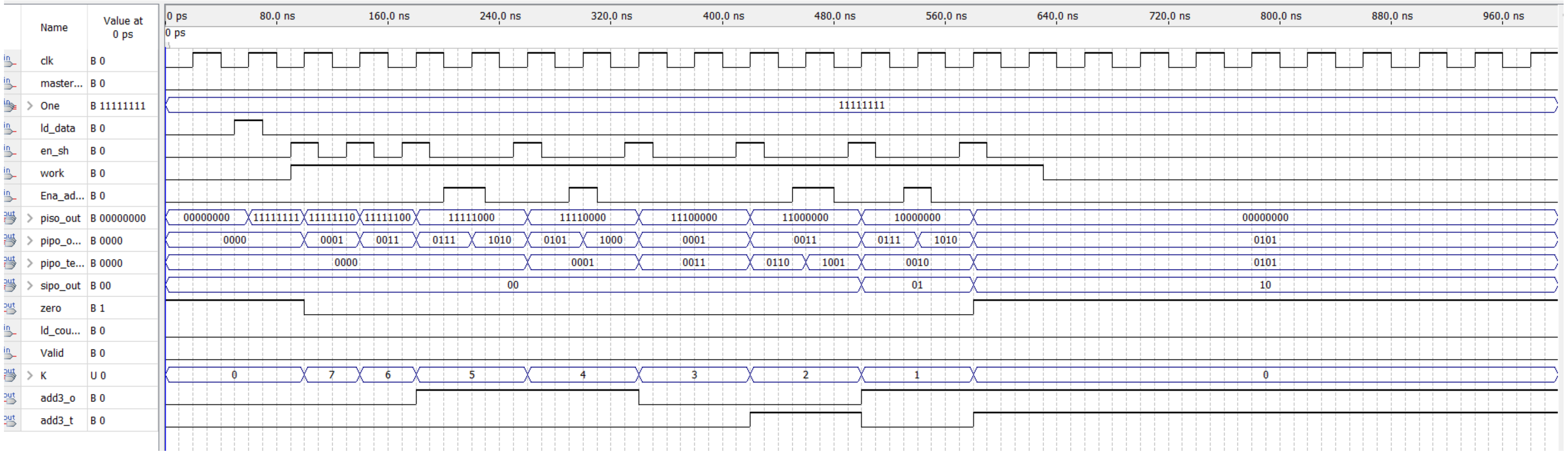
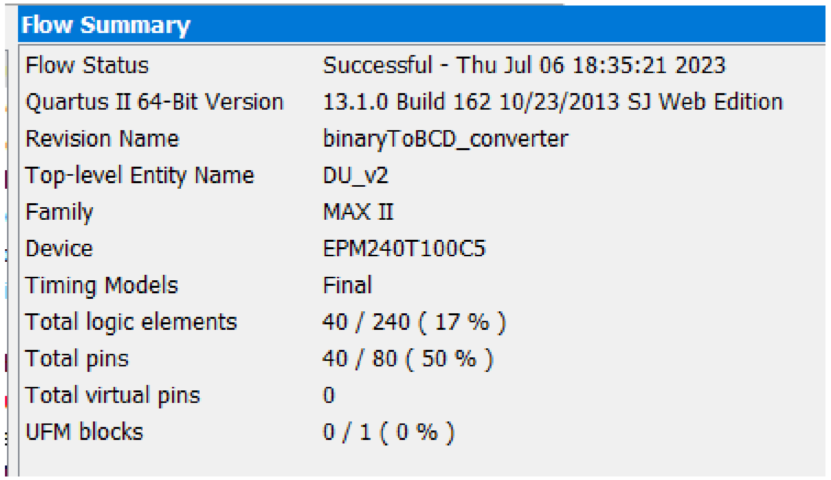


Figure 24: Waveform generated for DU



Flow Summary	
Flow Status	Successful - Thu Jul 06 18:35:21 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	DU_v2
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	40 / 240 (17 %)
Total pins	40 / 80 (50 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

Figure 25: Compilation Report of DU

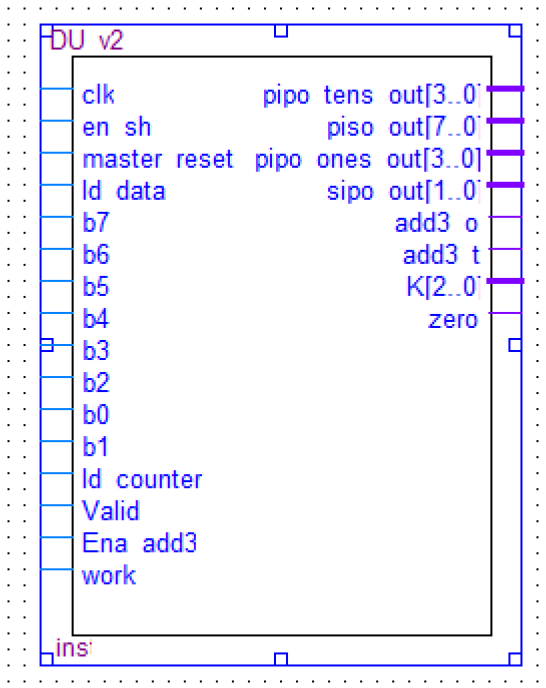


Figure 26: Symbol Diagram of DU

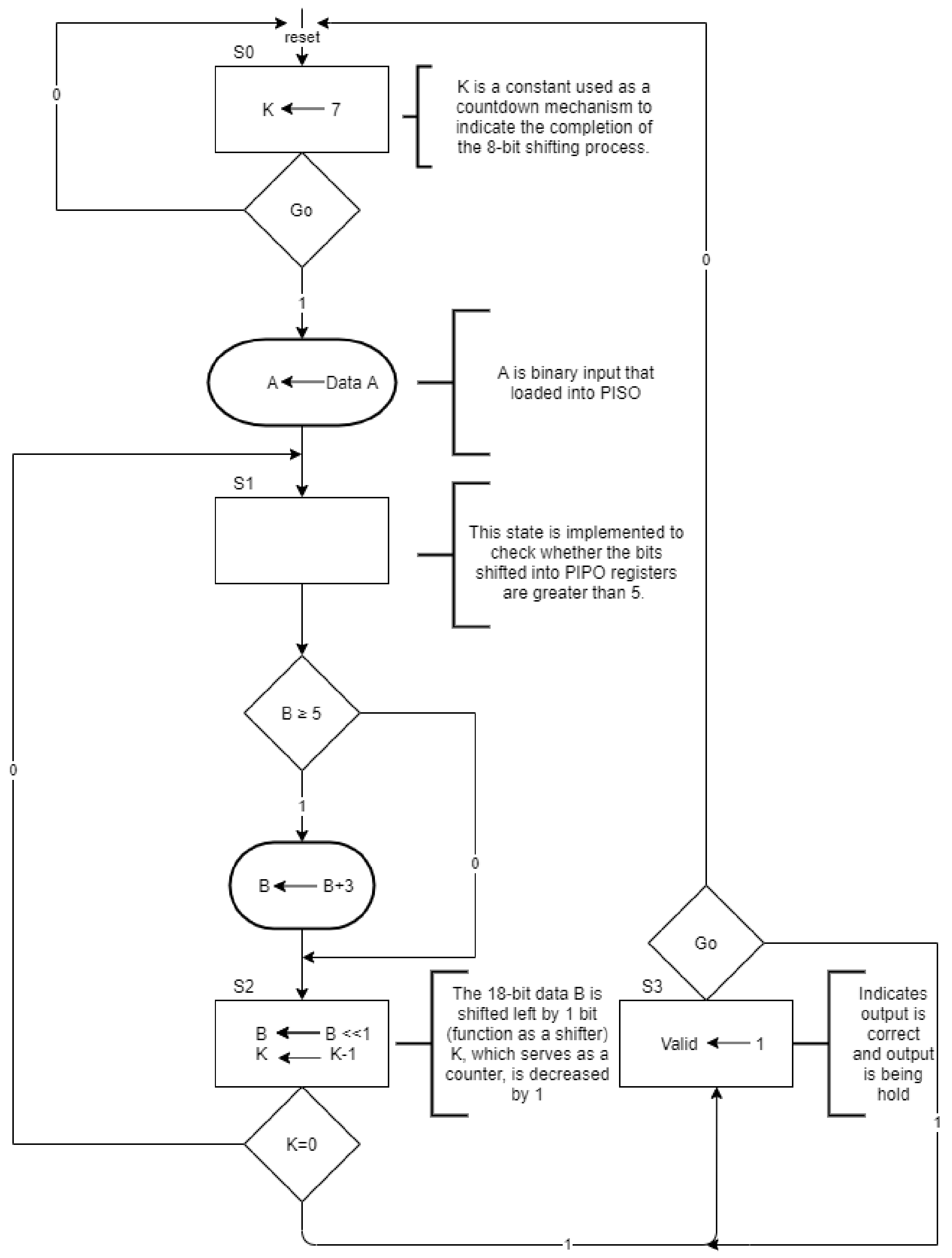
MARKING RUBRIC: M6

Group Name	Group 2	Section	9 & 10
Team Member 1	Titus Tong Zhong Xu		
Team Member 2	Chieng You Wei		
Team Member 3	Sim Zhong Xian		

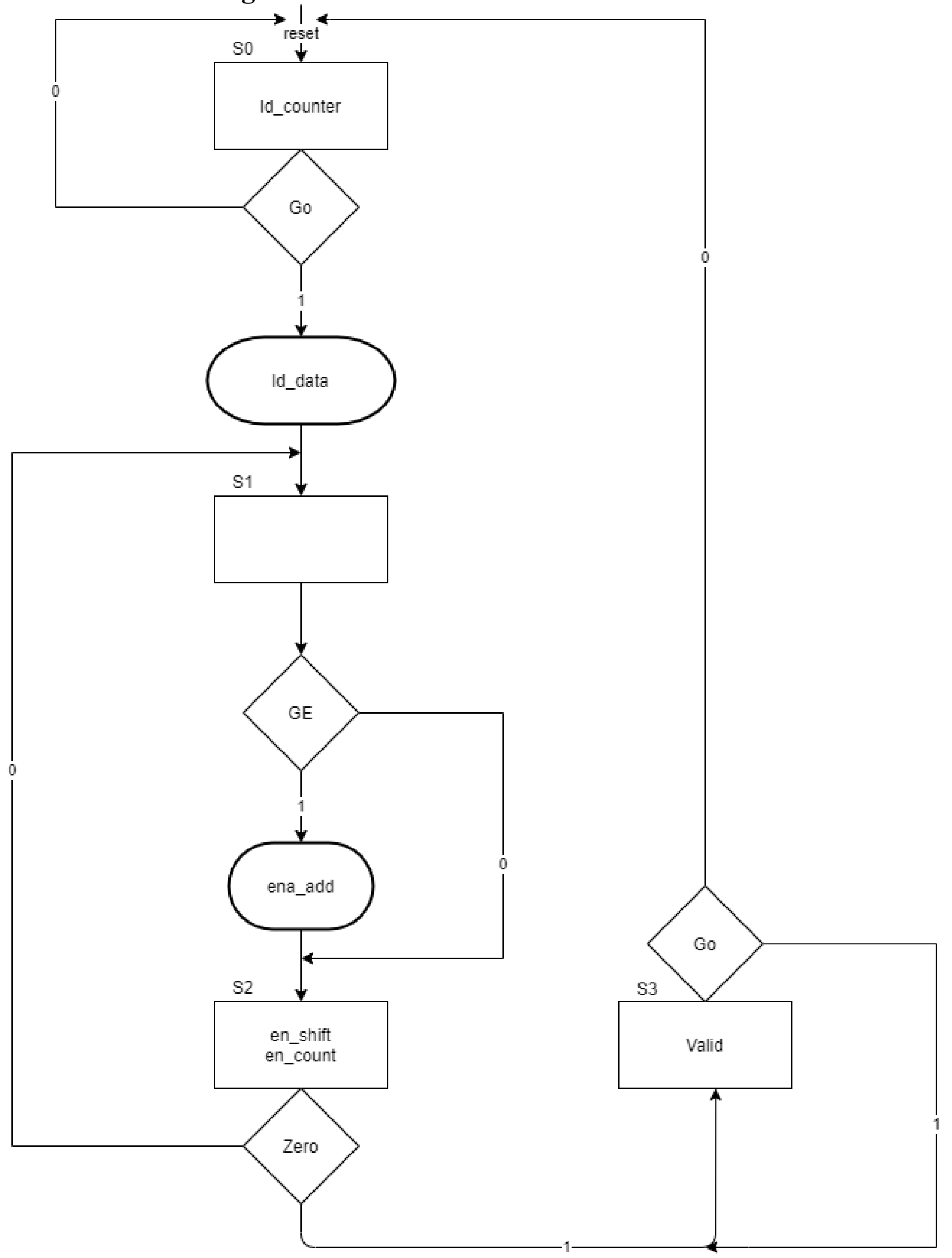
Item	Category	0	1	2	3	Score
1.	**Top-Down Design**: Approaches problem at high-level, then breaks the problem into smaller, more manageable pieces	Fails to grasp the concept of top-down design.	Understands the concept of top-down design.	Somewhat understands the concept of top-down design.	Fully understands and implements the full top-down design procedure and abstraction.	
2.	**Bottom-Up Implementation**: Start with the lowest level, ends with the top-level circuit	Seemingly random collections of circuits.	Mostly correct sequence.	Mostly incorrect sequence.	Bottom-up sequence is clearly evident.	
3.	**Documentation / Schematics**: Complete & clear. Proper signal labels.	Missing diagrams.	Hard to read diagrams. Improper use of labels.	Mostly complete.	Complete & clear documentation.	
4.	**Simulation Analysis**: Brief & direct to the point	Missing analysis.	Unsure what is important.	Reasonably easy to read, with minimum errors	Clear & easy to read reporting. Important results are highlighted.	
5.	**Test Data**: Covers all important test cases with no redundant cases	Missing simulation or no hardware.	Too few test cases and/or too many repeated test cases.	Covers most test cases and/or some repeated inputs.	Proper choice of test inputs.	
6.	**Hardware Functionality**: Working circuit	No hardware or hardware does not work.	Works but not to specifications.	Works according to specifications, sometimes.	Works flawlessly.	$\times 4$
7.	**Hardware Presentation**: Easy on the eyes.	No hardware.	A jumble of wires.	Acceptably neat.	Very neat.	
					Raw Marks	
					Final Assignment Marks ($\frac{Raw}{30} \times 5$)	

THE PLAGIARISM DECLARATION MUST BE SIGNED BY ALL TEAM MEMBERS AND THERE MUST NOT BE ANY FORM OF PLAGIARISM TO RECEIVE ANY MARKS.

Low Level ASM Chart


reset
S0
K ← 7
0
Go
1
K is a constant used as a countdown mechanism to indicate the completion of the 8-bit shifting process.
A ← Data A
A is binary input that loaded into PISO
S1
This state is implemented to check whether the bits shifted into PIPO registers are greater than 5.
B ≥ 5
1
0
B ← B+3
S2
B ← B <<1
K ← K-1
The 18-bit data B is shifted left by 1 bit (function as a shifter) K, which serves as a counter, is decreased by 1
K=0
1
0
S3
Valid ← 1
Go
Indicates output is correct and output is being hold
1
0

High Level ASM Chart



State Table (Binary Encoding)

Present State		Present Input			Next State		Present Output					
Q_1	Q_0	Go	Zero	GE	Q_1^+	Q_0^+	ld_counter	en_shift	en_count	Valid	ena_add	ld_data
00		0	0	0	0	0	1	0	0	0	0	0
		0	0	1	0	0	1	0	0	0	0	0
		0	1	0	0	0	1	0	0	0	0	0
		0	1	1	0	0	1	0	0	0	0	0
		1	0	0	0	1	1	0	0	0	0	1
		1	0	1	0	1	1	0	0	0	0	1
		1	1	0	0	1	1	0	0	0	0	1
		1	1	1	0	1	1	0	0	0	0	1
01		0	0	0	1	0	0	0	0	0	0	0
		0	0	1	1	0	0	0	0	0	1	0
		0	1	0	1	0	0	0	0	0	0	0
		0	1	1	1	0	0	0	0	0	1	0
		1	0	0	1	0	0	0	0	0	0	0
		1	0	1	1	0	0	0	0	0	1	0
		1	1	0	1	0	0	0	0	0	0	0
		1	1	1	1	0	0	0	0	0	1	0

10	0	0	0	0	1	0	1	1	0	0	0
	0	0	1	0	1	0	1	1	0	0	0
	0	1	0	1	1	0	1	1	0	0	0
	0	1	1	1	1	0	1	1	0	0	0
	1	0	0	0	1	0	1	1	0	0	0
	1	0	1	0	1	0	1	1	0	0	0
	1	1	0	1	1	0	1	1	0	0	0
	1	1	1	1	1	0	1	1	0	0	0
11	0	0	0	0	0	0	0	0	1	0	0
	0	0	1	0	0	0	0	0	1	0	0
	0	1	0	0	0	0	0	0	1	0	0
	0	1	1	0	0	0	0	0	1	0	0
	1	0	0	1	1	0	0	0	1	0	0
	1	0	1	1	1	0	0	0	1	0	0
	1	1	0	1	1	0	0	0	1	0	0
	1	1	1	1	1	0	0	0	1	0	0

Next State Equations:

$$S_0^+ = S_0\overline{GO} + S_3\overline{GO}$$

$$S_1^+ = S_0GO + S_2\overline{ZERO}$$

$$S_2^+ = S_1$$

$$S_3^+ = S_2ZERO + S_3GO$$

Output Equations:

$$ld\_counter = S_0$$

$$ld\_data = S_0GO$$

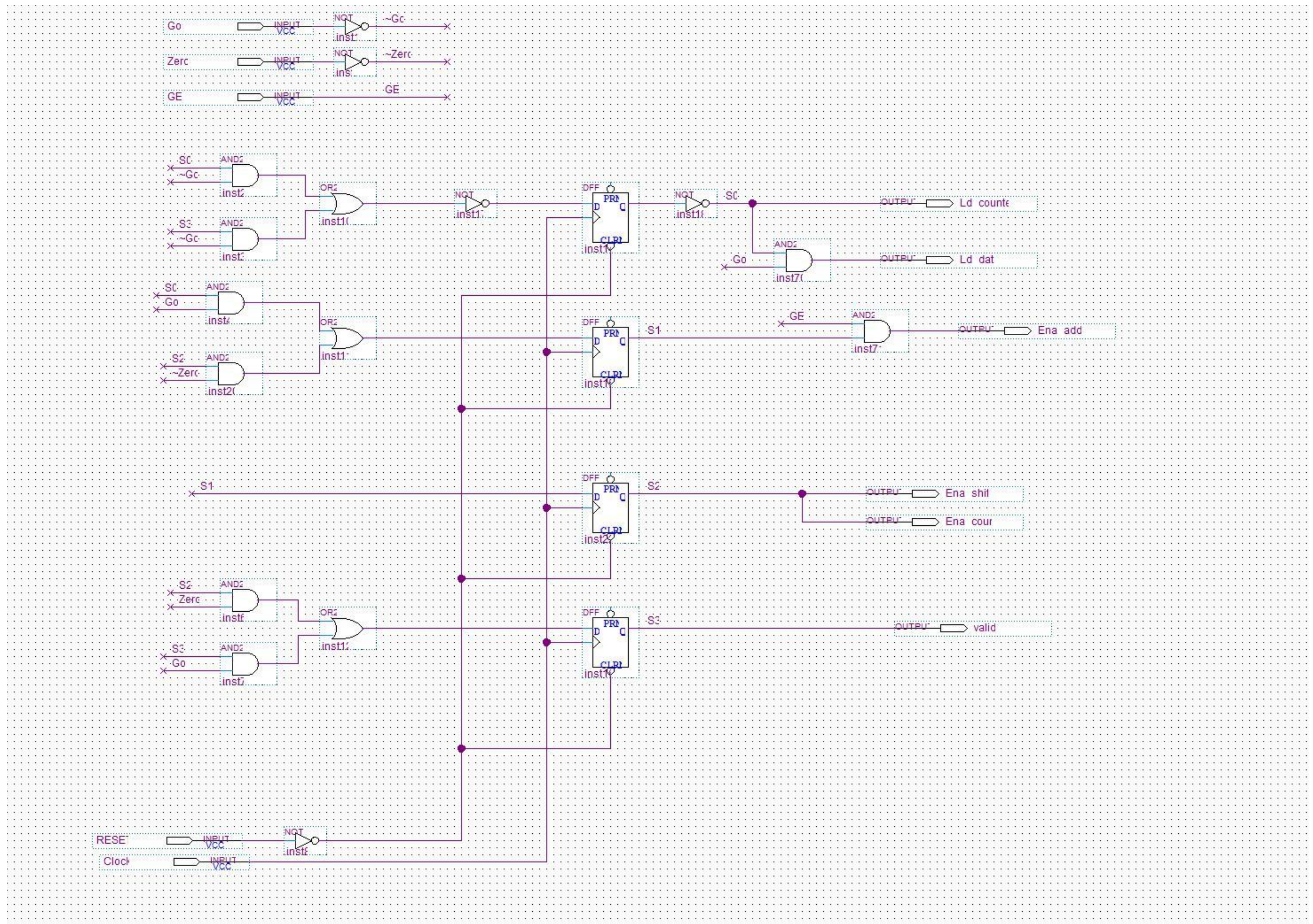
$$ena\_add = S_1GE$$

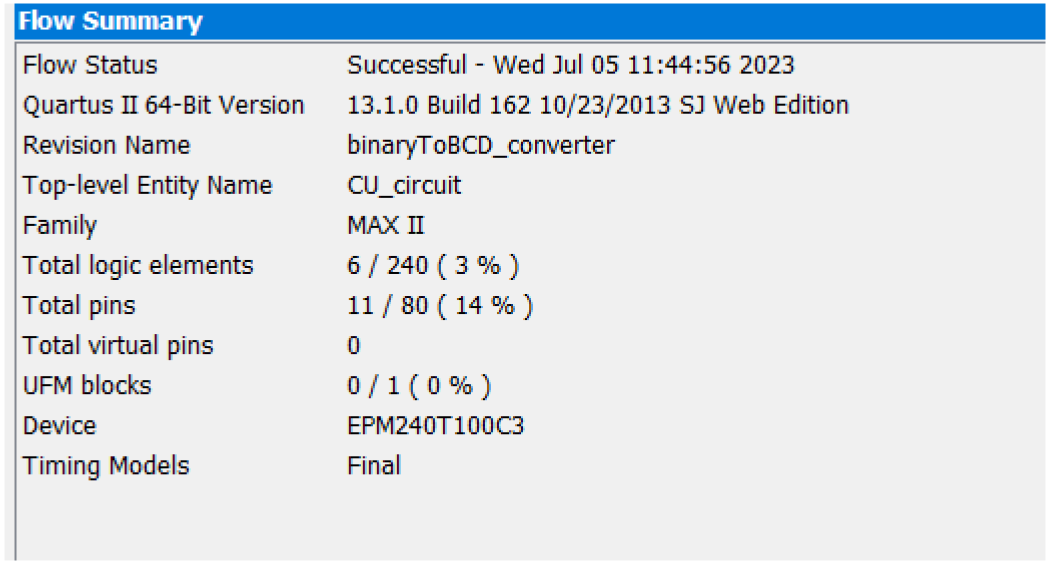
$$en\_shift = S_2$$

$$en\_count = S_2$$

$$Valid = S_3$$

Logic Circuit





Flow Summary

Flow Status	Successful - Wed Jul 05 11:44:56 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	CU_circuit
Family	MAX II
Total logic elements	6 / 240 (3 %)
Total pins	11 / 80 (14 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)
Device	EPM240T100C3
Timing Models	Final

Figure 1: Compilation report

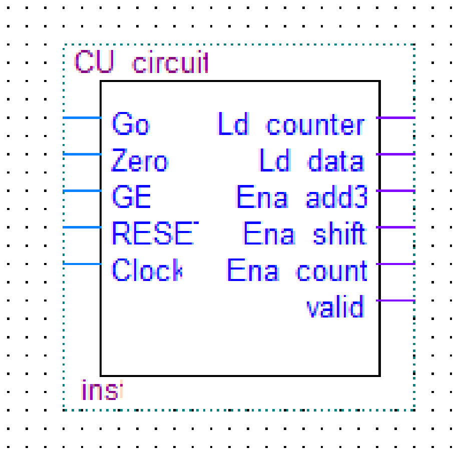
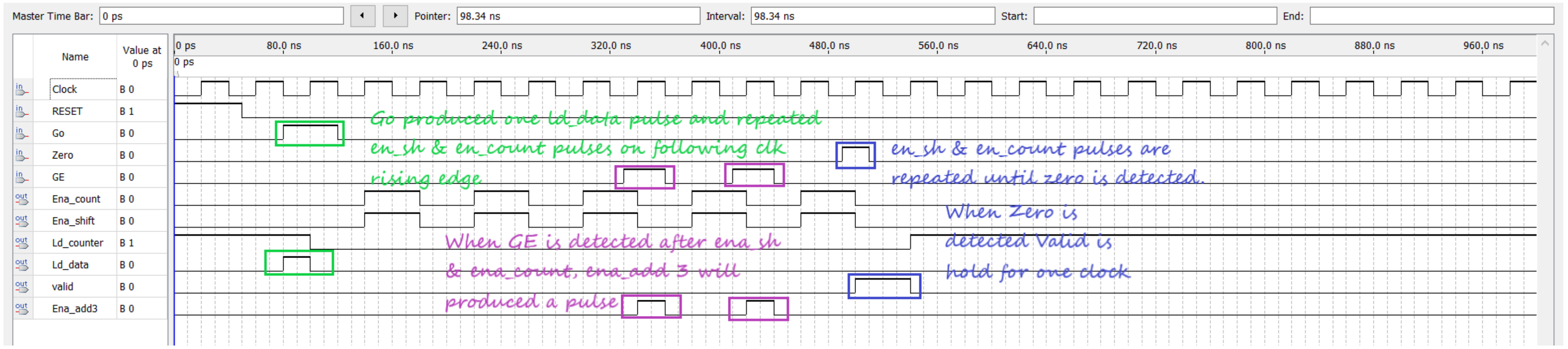


Figure 2: Symbol for CU



Master Time Bar: 0 ps Pointer: 98.34 ns Interval: 98.34 ns Start: End:

Name	Value at 0 ps
Clock	B 0
RESET	B 1
Go	B 0
Zero	B 0
GE	B 0
Ena_count	B 0
Ena_shift	B 0
Ld_counter	B 1
Ld_data	B 0
valid	B 0
Ena_add3	B 0

Go produced one ld_data pulse and repeated en_sh & en_count pulses on following clk rising edge.

en_sh & en_count pulses are repeated until zero is detected.

When GE is detected after ena_sh & ena_count, ena_add 3 will produced a pulse

When Zero is detected Valid is hold for one clock

Figure 3: Annotated simulation output waveform

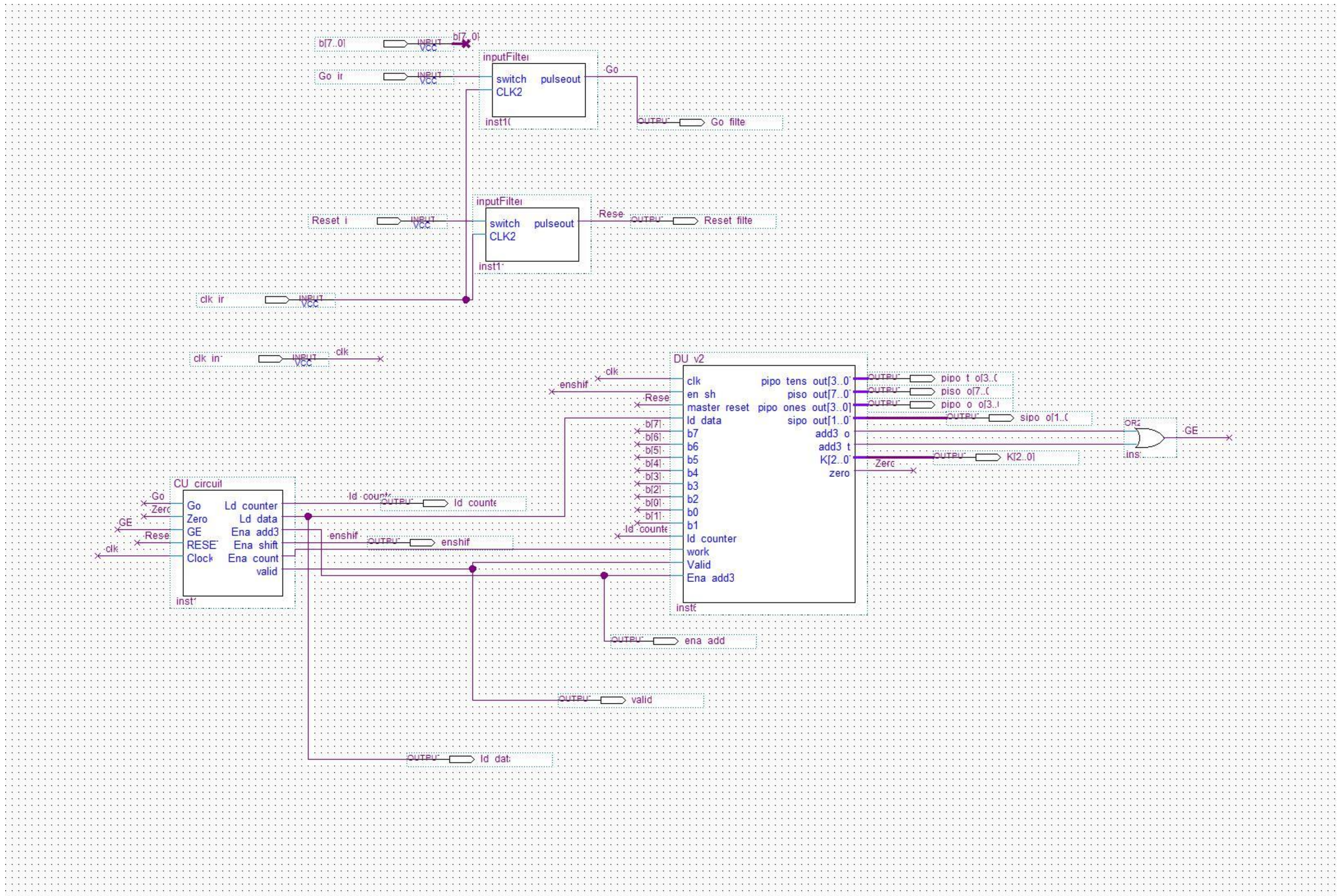
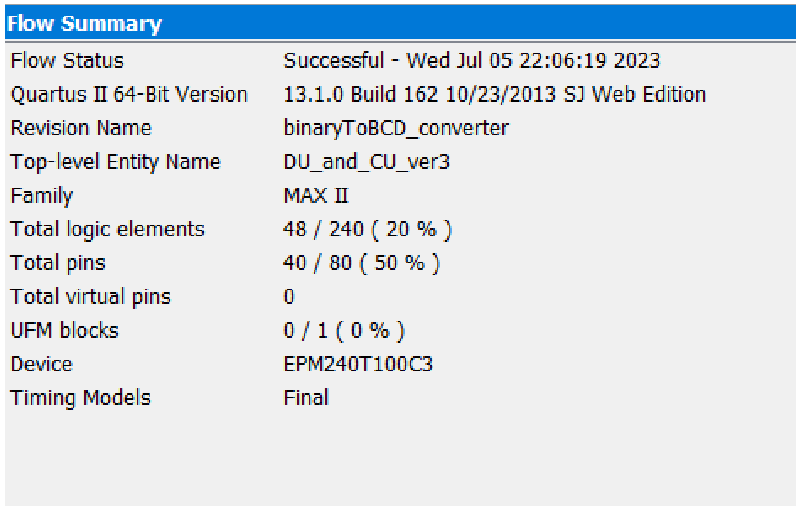


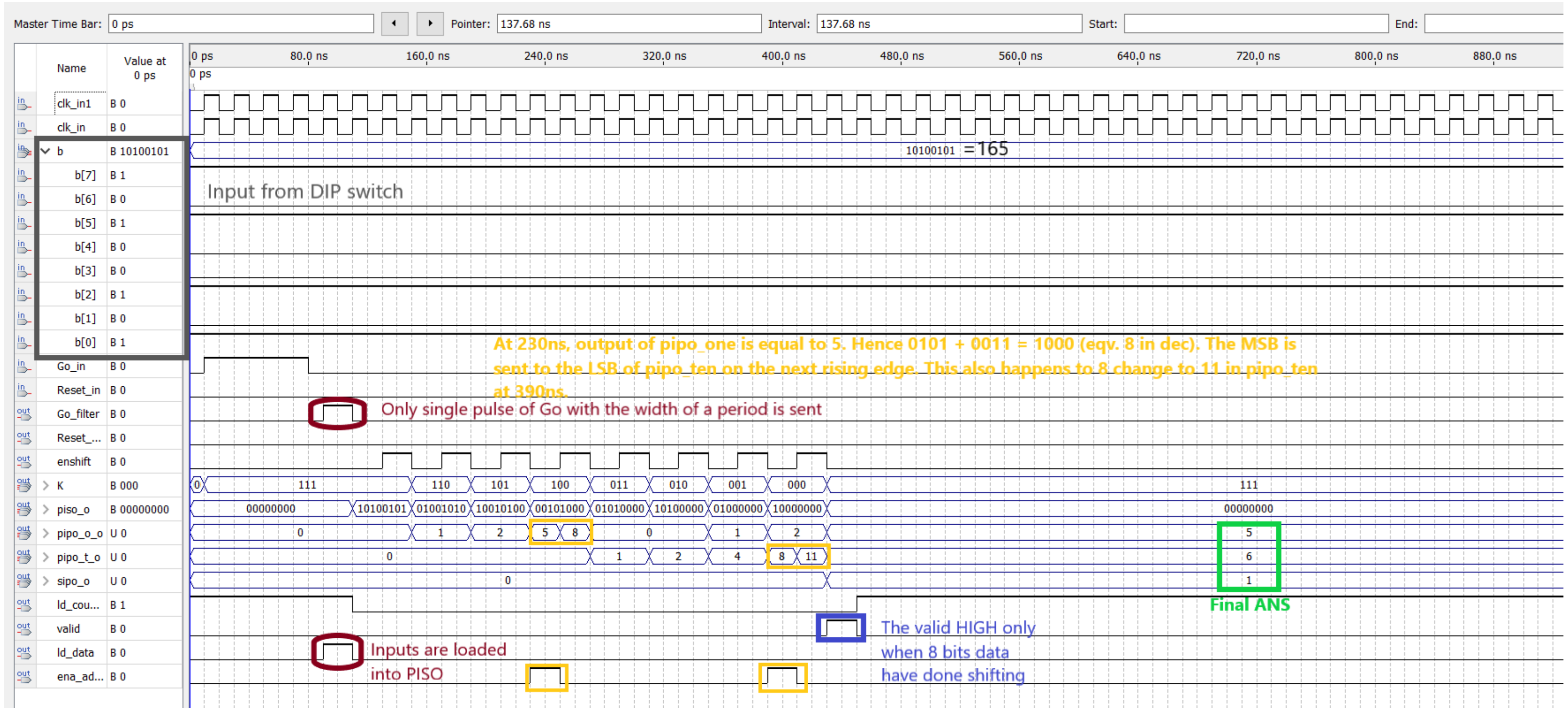
Figure 4: DU connected with CU circuit schematic



Flow Summary

Flow Status	Successful - Wed Jul 05 22:06:19 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	binaryToBCD_converter
Top-level Entity Name	DU_and_CU_ver3
Family	MAX II
Total logic elements	48 / 240 (20 %)
Total pins	40 / 80 (50 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)
Device	EPM240T100C3
Timing Models	Final

Figure 5: Compilation report of CU and DU circuit



Input from DIP switch

At 230ns, output of pipo_one is equal to 5. Hence 0101 + 0011 = 1000 (eqv. 8 in dec). The MSB is sent to the LSB of pipo_ten on the next rising edge. This also happens to 8 change to 11 in pipo_ten at 390ns.

Only single pulse of Go with the width of a period is sent

Inputs are loaded into PISO

The valid HIGH only when 8 bits data have done shifting

Figure 6: CU and DU simulated output waveform

Reference

Video Link: https://youtu.be/6RjLzIGnJbU

AN08 Simple Input:Output with CPLD. Retrieved from
AN08 Simple Input:Output with CPLD.pdf

AN09 Using BCD-to-Seven-Segment Decoders Drivers. Retrieved from
AN09 Using BCD-to-Seven-Segment Decoders Drivers.pdf

Munim, P. M. (2023). Cost-Effective Digital System Construction Manual.

Munim, Kamisian, & I. (2022). The Art of Digital System Design (2022 ed.).

Thomas L. Floyd, "Digital Fundamentals" 10th Ed., Prentice Hall, 2009.