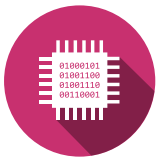


Arithmetische und Logische Einheit (ALU)

Labor Digital Design

Inhalt

1 Ziel

Dieses Labor dient, den Entwurf von logischen Schaltungen mit Hilfe von Multiplexern zu üben. Es zeigt eine Methode zur Erstellung von arithmetischen und logischen Einheiten für Mikroprozessore.

In diesem Labor wird eine Methode zur Realisierung einer Arithmetischen und Logischen Mikroprozessoreinheit vorgestellt. Die Logik- und Arithmetikeinheiten werden in einer ersten Laborsitzung realisiert, während die komplette Arithmetic and Logical Unit (ALU) in einer zweiten Sitzung fertiggestellt wird.

2 Logische Einheit LU

Die Abbildung 1 zeigt die Schaltung einer Logischen Einheit (Logical Unit (LU)) eines Mikroprozessors. Die logische Operationen werden Bit für Bit durchgeführt.

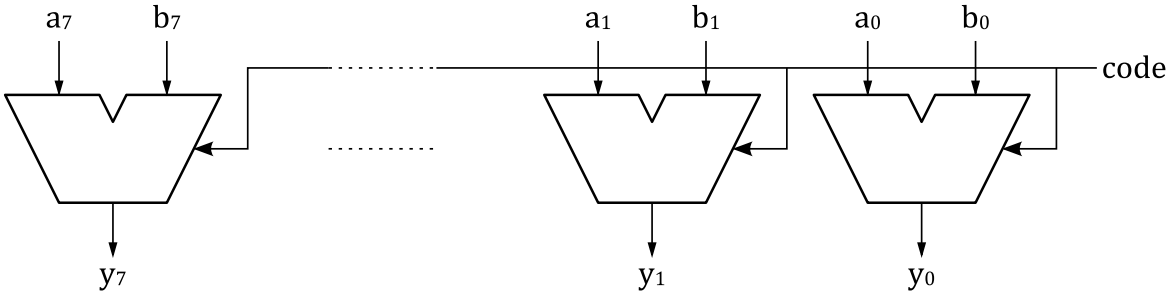


Abbildung 1: Logische Einheit

Die Iterativblöcke der **(abbr: [LU], long: [Logical Unit])** werden mit Multiplexern erstellt, welche eine Wahrheittabelle erstellen, wobei die Steuereingänge $\text{sel}_0 = a_i$, $\text{sel}_1 = b_i$ und $(\text{sel}_3, \text{sel}_2) = \text{`code}[1:0]$ zur Bestimmung der zu erzeugenden Funktion dienen.

Schreiben Sie die Wahrheitstafel der Logikfunktion, welche folgende Operationen im programmierbaren Logikblock erzeugt.

- $y_i = b_i$ für **`code = "00"`** Laden von b
- $y_i = a_i * b_i$ für **`code = "01"`** UND Funktion zwischen a und b
- $y_i = a_i + b_i$ für **`code = "10"`** ODER Funktion zwischen a und b
- $y_i = a_i \oplus b_i$ für **`code = "11"`** exklusiv-ODER Funktion zwischen a und b

2.1 Erstellung und Simulation

Vervollständigen Sie die Schaltung des Iterativblocks der Logical Unit, welche die 4 angegebenen Operationen durchführt. Vervollständigen Sie die Teststimuli und überprüfen Sie die Funktion der gesamten LU .

3 Arithmetische Einheit AU

Die Abbildung 2 zeigt die Schaltung einer Arithmetischen Einheit (Arithmetic Unit (AU)).

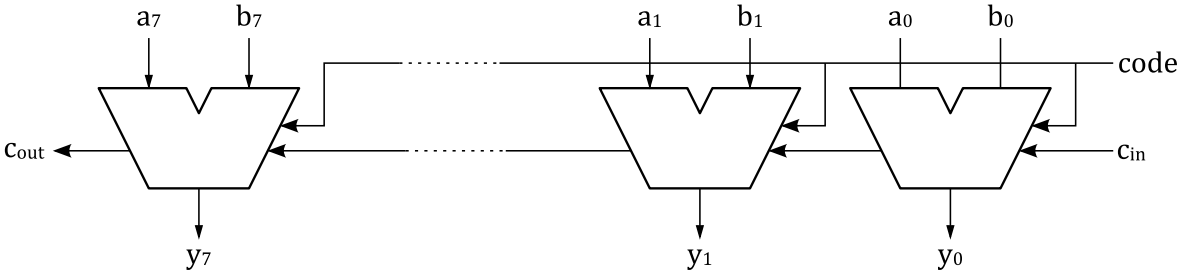


Abbildung 2: Arithmetische Einheit

Schreiben Sie die Wahrheitstafel des iterativen Logikblocks, welcher dazu dient, die folgenden Operationen auf ganze Zahlen zu ermöglichen.

- $y = a + b$ pour **`code[0] = '0'`** ⇨ Addition
- $y = a - b$ pour **`code[0] = '1'`** ⇨ Subtraktion

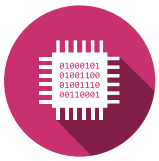
Die Funktion wird erstellt durch einen Multiplexer, dessen Steuereingänge $\text{sel}_0 = a_i$, $\text{sel}_1 = b_i$, $\text{sel}_2 = c_{\text{in}}$ und $\text{sel}_3 = \text{code}[0]$ sind.

Mit der Ansicht, dass eine Schiebung nach links eine Multiplikation mit 2 entspricht, schlagen Sie eine Änderung der iterativen Schaltung vor, um die Schiebeoperation nach links : $y = a \ll 1 = a + a$ für **`code[1] = '1'`** zu erzeugen.

3.1 Erstellung und Simulation

Vervollständigen Sie die Schaltung des Iterativblocks der **`(abbr: [AU], long: [Arithmetic Unit])`**, welche die 3 angegebenen Operationen durchführt. Vervollständigen Sie die Teststimuli und überprüfen Sie die Funktion der gesamten AU .

> ⚠ Stellen Sie sicher das Sie alle Funktionen und Möglichkeiten vollständig testen.



4 Arithmetische und logische Einheit

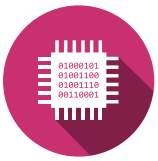
4.1 Auswahl der Resultate

Die arithmetische und logische Einheit (Arithmetic and Logical Unit (ALU)) wir hier erstellt durch die Zusammensetzung der LU und der AU, welche hierfür entworfen wurden. Die ALU wir hier erstellt durch die Zusammensetzung der LU und der AU , welche hierfür entworfen wurden. Die ALU enthält dazu noch eine Schiebeoperation nach rechts.

Ergänzen Sie die Schaltung der ALU, um die Steuersignale der Multiplexer zu generieren, um die richtige Operationen zu selektieren. Diese Operationen sind in der Tabelle Tabelle 1

Instruction $I_{17} : I_{13}$	Mnemonic	ALU code Operation	Operation
00000	LOAD	LOAD B	$y = b$
00001	*ununsed*	-	-
00010	INPUT	LOAD B	$y = b$
00011	FETCH	LOAD B	$y = b$
00100	*ununsed*	-	-
00101	AND	AND	$y = a$ AND b
00110	OR	OR	$y = a$ OR b
00111	XOR	XOR	$y = a$ XOR b
01000	*ununsed*	-	-
01001	TEST	AND	$y = a$ AND b
01010	COMPARE	SUB	$y = a - b$
01011	*ununsed*	-	-
01100	ADD	ADD	$y = a + b$
01101	ADDCY	ADDCY	$y = a + b + c_{\text{in}}$
01110	SUB	SUB	$y = a - b$
01111	SUBCY	SUBCY	$y = a - b - c_{\text{in}}$
10000	SH / ROT	SHR	$a \gg 1$
10001	SH / ROT	SHL	$a \ll 1$
10010	*ununsed*	-	-
10011	*ununsed*	-	-
10100	*non-ALU*	-	-
...	...	...	...
11111	*non -LU*	-	-

Tabelle 1: ALU-Befehlssatz



4.2 Erstellung und Simulation

Abhand der Tabelle 1, füllen Sie die Wahrheitstabelle der Tabelle 2, welche die Steuersignale der Multiplexer und diejenige der LU und der AU als Funktion des ALU codes (code[4 : 0]) angibt.

code[4 : 0]	$LU_{code}[1:0]$	$AU_{code}[1:0]$	$select_{AU}$	$select_{SR}$	c_{in_AU}
00000					
00001					
00010					
00011					
00100					
00101					
00110					
00111					
01000					
01001					
01010					
01011					
01100					
01101					
01110					
01111					
10000					
10001					
10010					
10011					
10100					
...					
11111					

Tabelle 2: ALU-Steuerungen

Schreiben Sie die Gleichungen dieser Steuersignale.

Vervollständigen Sie die Schaltung der ALU und dessen Stimuli. Überprüfen Sie die Funktionalität der ALU.

Stellen Sie sicher das Sie alle Funktionen und Möglichkeiten vollständig testen.