Universidad del Valle
Facultad de Ingeniería, Escuela EIEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2:30 horas

Feb. 5 del 2016

1. Diseñar la unidad de control para realizar la raíz cuarta en el procesador UV-ARM. *ASM (10%)* y prueba de escritorio: dato = 101111(5%)

2. Diseñar *data-path* específico y la unidad de control para calcular X^4. *ASM* (10%) y prueba de escritorio: dato = 1010 (5%)

3. Diseñar un circuito controlador para realizar la división N/N bits. El dividendo se encuentra en R0, y el divisor en R1 del banco de registros del procesador UV-ARM7 de la Figura 1. El cociente y el residuo deben ser almacenados en R7 y R8, respectivamente. *RTL-ASM* (15%) y prueba de escritorio. Dividendo = 110111 y divisor = 000111 (5%).

4. Diseñar un circuito controlador para realizar la multiplicación basada en el algoritmo de Booth. El multiplicando se encuentra en R0, y el multiplicador en R1 del banco de registros del procesador UV-ARM7 de la Figura 1. El producto debe ser almacenado en R3 y R4. *ASM* (20%) y prueba de escritorio: multiplicando = 1011 y multiplicador = 1101 (5%).

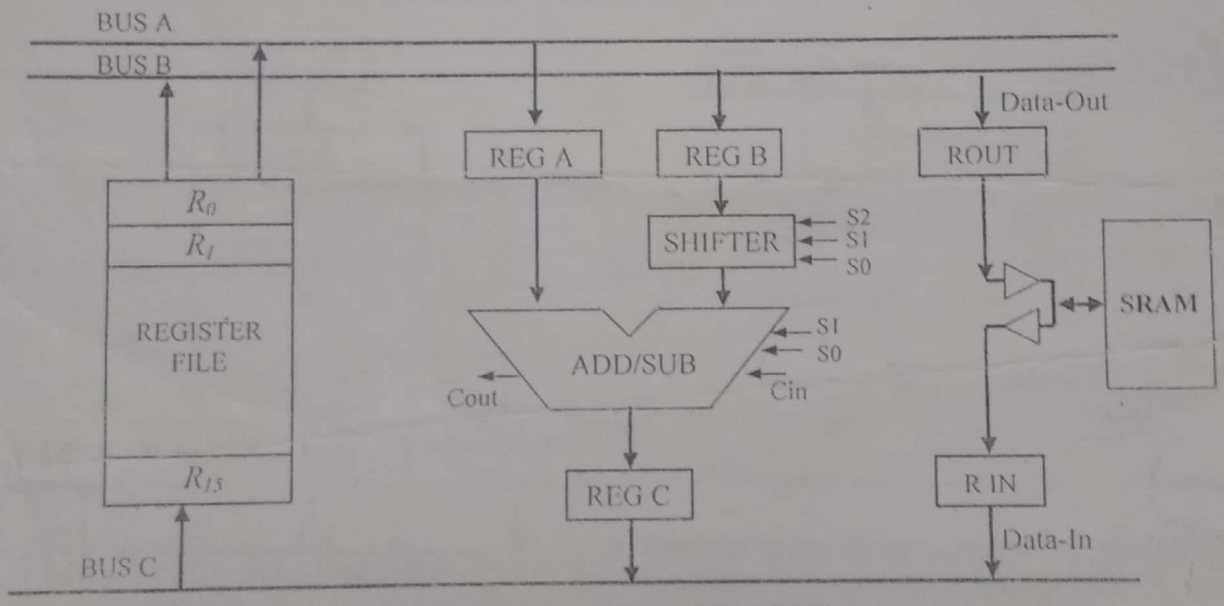


Figura 1: Procesador UV-ARM7

5. Diseñar un circuito controlador para realizar la "*multiplicación*" usando el algoritmo de **Radix 8** (multiplica 3-bits simultáneamente), los datos están en R1 (multiplicando) y R2 (multiplicador), y el producto debe ser almacenado en R7 (parte alta) y R8 (parte baja) del procesador UV-ARM7 de la Fig. 1. RTL-*ASM* (25%)
Ejemplo: multiplicador 101010 y multiplicando 101010:

Nota:
- ALU (S_1S_0): 00 = A or B ; 01= A and B ; 10 = A+B ; 11 = A-B
- Shifter (S2S1S0): 000 = lógico-left {0} ; 001 = lógico-left {1} ; 010 = lógico-right {0} ; 011 = lógico-right{1}
- 1XX = transferir
- Los ASMs solo tendrán tres notas: 5.0 para bueno y óptimo (un número mínimo de estados); 2.5 para bueno pero no óptimo y 1.0 para diagrama malo.

Parcial 2

Universidad del Valle
Facultad de Ingeniería, Escuela IEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2 horas y 30 minutos

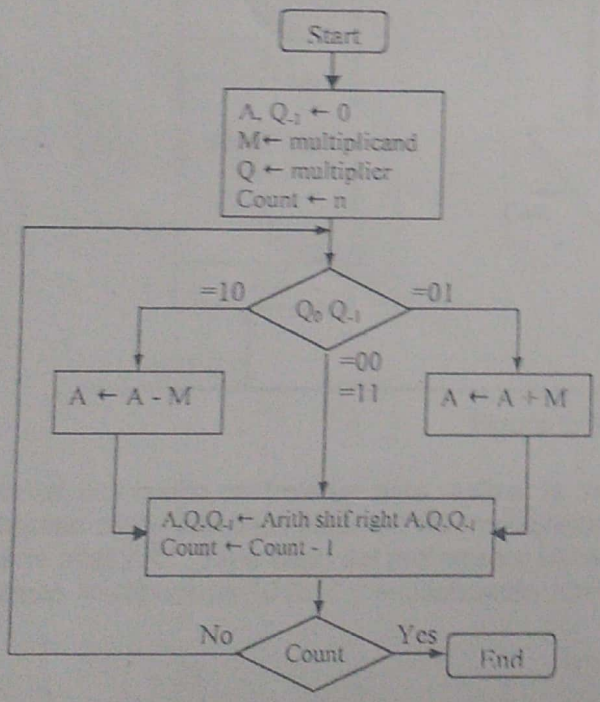
Junio 4 del 2014

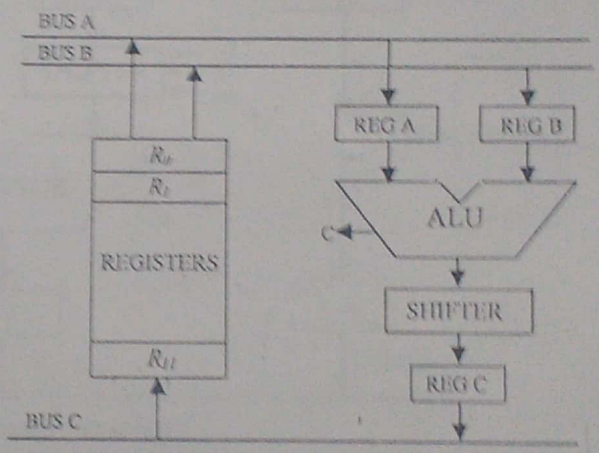
1. Diseñar un circuito detector de secuencias: (15%)
-Si detecta 00,11,10 la salida es 01 y permanece en 01 hasta detectar 10,00,10, que lleva a la salida a 00
-Si detecta 10,00,01 la salida es 10 y permanece en 10 hasta detectar 01,11,01, que lleva a la salida a 00
-Existe superposición, en este caso, si la salida está en 01 puede pasar a 10 si detecta la respectiva secuencia, y viceversa. La FSM debe ser Mealy

2. Diseñar una FSM Mealy síncrona para controlar un motor DC: (15%).
- El motor arranca y gira en sentido normal cuando detecta la secuencia 00110; salida Z1Z0 = 01
- El motor arranca y gira en sentido contrario cuando detecta la secuencia 11001; salida Z1Z0 = 10
- El motor se detiene cuando detecta la secuencia 1010; salida Z1Z0 = 00
- El motor gira en sentido contrario cuando detecta la secuencia 0101 pero primero debe parar antes de cambiar de giro; salida Z1Z0 = 00*.

3. Diseñar un data-path específico y la unidad de control para realizar división. Data-path y ASM (15%) y prueba de escritorio: dividendo = 11010101 y divisor 0001010l(5%)

4. Diseñar la unidad de control para realizar la operación $1/X^2$. El dato X (8-bits) está en R1, el resultado debe ser almacenado en R2 del procesador UV3. ASM (25%)





5. Realizar el control para realizar la multiplicación en el procesador UV3 usando el algoritmo de Booth radix-2. ASM (20%)

Qi	Qi-1	Operation
0	0	Shift only
1	1	Shift only
1	0	Subtract and shift
0	1	Add and shift

Ejemplo: Multiplicador Q= 1010, codificación: 00; 10; 01; 10
Realizar la prueba de escritorio para el ejemplo: 8x(-9)=72 (5%)

Nota: ALU→ 00: Transferir A; 01: Sumar A + B; 10: Restar A – B; 11: Función A and B
Shifter→ 1XX: no shift ; 000: logical shift left, 0 ; 001: logical shift right,0 ; 010: arithmetic shift right ; 011: shift left,1
Los diagramas de estado y ASM solo tendrán tres notas: 5.0 (óptimo); 3.5 (bueno, pero no óptimo) y 1.0 (malo)

Segundo Parcial

Universidad del Valle
Facultad de Ingeniería, Escuela EIEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2:30 horas

Nov. 22 del 2016

1. Diseñar *data-path* específico y la unidad de control para calcular $X^{1/4}$. *ASM* (15%) y prueba de escritorio: dato = 101010 (10%)

2. Diseñar un circuito controlador para realizar la división N/N bits en el procesador UV-ARM de la Figura 1. El dividendo se encuentra en R2, y el divisor en R3 del banco de registros. El cociente y el residuo deben ser almacenados en R4 y R5, respectivamente. *RTL-ASM* (20%) y prueba de escritorio: Dividendo = 110111 y divisor = 000111 (5%).

3. Diseñar un circuito controlador para realizar la multiplicación basada en el algoritmo de Booth radix-2. El multiplicando se encuentra en R2, y el multiplicador en R3 del banco de registros del procesador UV-ARM7 de la Figura 1. El producto debe ser almacenado en R4 y R5. *ASM* (20%) y prueba de escritorio: multiplicando = 1101 y multiplicador = 1101 (5%).

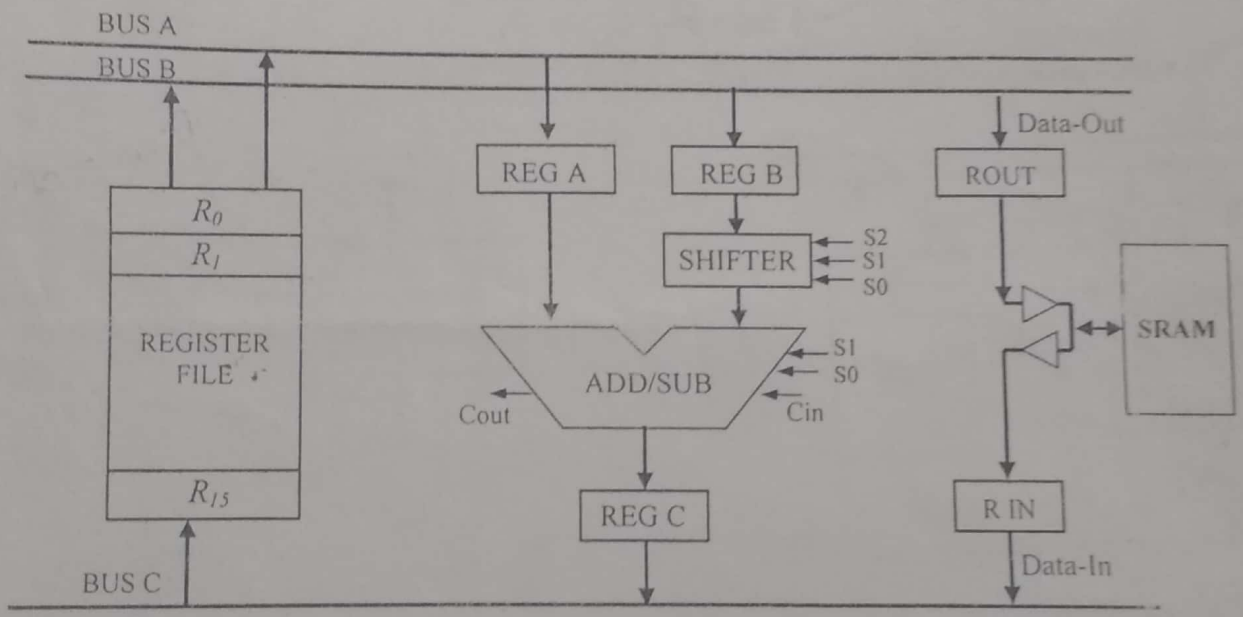


Figura 1: Procesador UV-ARM7

4. Diseñar un circuito controlador para realizar la "*multiplicación*" usando el algoritmo de **Radix 8** (multiplica 3-bits simultáneamente), los datos están en R0 (multiplicando) y R1 (multiplicador), y el producto debe ser almacenado en R6 (parte alta) y R7 (parte baja) del procesador UV-ARM7 de la Fig. 1. RTL-*ASM* (25%)
Ejemplo: multiplicador 101010 y multiplicando 101010:

Nota:
ALU (S_1S_0): 00 = A or B ; 01= A and B ; 10 = A+B ; 11 = A-B
Shifter (S2S1S0): 000 = lógico-left {0} ; 001 = lógico-left {1} ; 010 = lógico-right {0} ; 011 = lógico-right{1}
1XX = transferir
Los ASMs solo tendrán tres notas: 5.0 para bueno y óptimo (un número mínimo de estados); 2.5 para bueno pero no óptimo y 1.0 para diagrama malo.

Universidad del Valle
Facultad de Ingeniería, Escuela EIEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2 horas

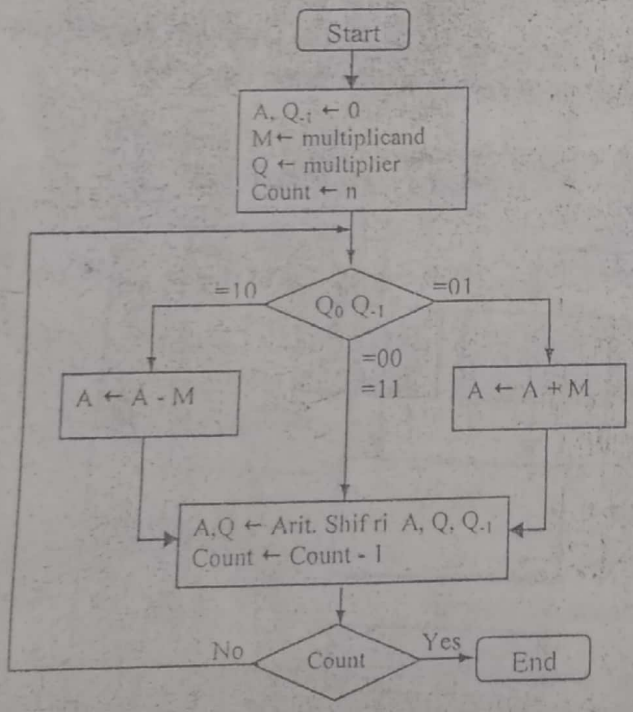
Junio 4 del 2010

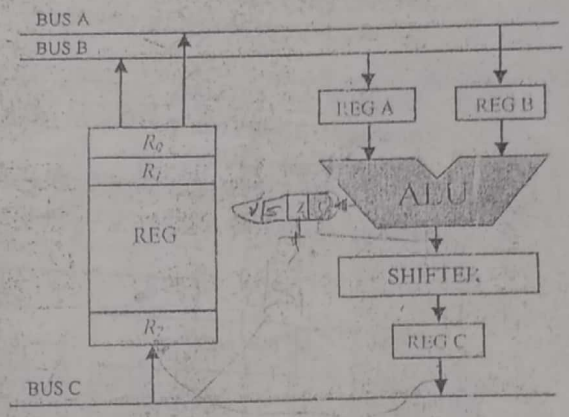
1. Diseñar una FSM para detectar dos secuencias: Diagrama de estados (20%)
- cuando se detecta la secuencia: X = 10110, las salidas Z1Z2 = 10 y las salidas permanecen en 10 hasta detectar la secuencia 10010 entonces Z1Z2 = 00
- cuando se detecta la secuencia: X = 01001 las salidas Z1Z2 = 01 y las salidas permanece en 01 hasta detectar la secuencia 01101 entonces Z1Z2 = 00
- las dos secuencias se pueden superponer, FSM Mealy

2. Diseñar el datapath y la unidad de control (FSM, diagrama de estados) para un circuito que permita realizar la función. (20%):

$$y(n)=\sum_{i=0}^{2} a_i y(n-i)+\sum_{k=0}^{2} b_k x(n-k)+\sum_{j=0}^{2} Cj$$

3. Diseñar un circuito controlador para realizar la división usando el UV2010. ASM (30%)
Dividendo = 10010011 (R0-R1) y divisor = 1011 (R2). Usar el algoritmo con restauracion



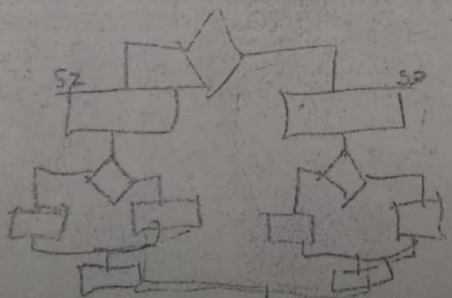


4. Realizar el control para realizar la multiplicación en UV2010 usando el algoritmo de Booth radix-2. ASM (30%)

Qi	Qi-1	Operation
0	0	Shift only
1	1	Shift only
1	0	Subtract and shift
0	1	Add and shift

Ejemplo: Multiplicador Q= 1010
Codificación: 00; 10; 01; 10

Nota:
- ALU→ 00: Transferir A; 01: Sumar A + B; 10: Restar A – B; 11: Funcion A and B
- Shifter→ 00: no shift ; 01: shift left,0 ; 10: shift left,1; 11: shift right,0
- Los diagramas de estado solo tendrán tres notas: 5.0; 2.5 y 1.0



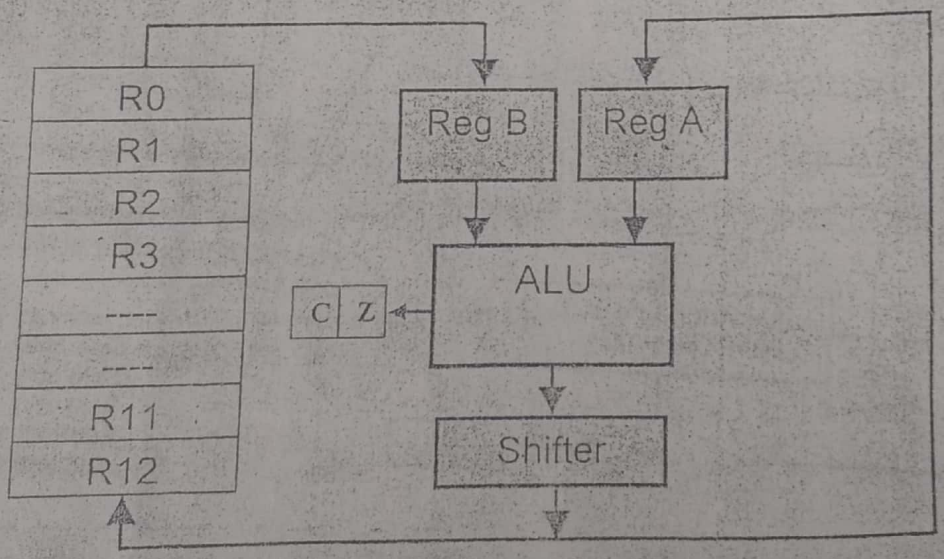
Mayo 31, 2016

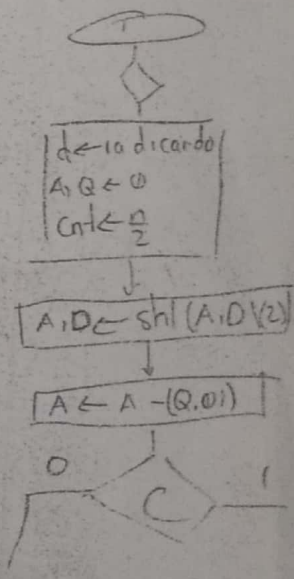
Universidad del Valle
Facultad de Ingeniería, Escuela EIEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina

1. Diseñar una FSM asíncrona para controlar un motor DC, existe superposición: Diagrama de estados, usar FSM tipo Mealy, (20%)
   - El motor arranca y gira en sentido normal cuando detecta la secuencia X1 X2 X1 X2
   - El motor arranca y gira en sentido contrario cuando detecta la secuencia X3 X2 X3 X1
   - El motor se detiene, si el circuito detecta la secuencia X3 X1 X2 y permanece detenido hasta detectar de nuevo la secuencia de arranque
   - Sin embargo, después de arrancar el motor, cada vez que el circuito detecta la secuencia X2 X3 X1 X3, el motor debe girar en sentido contrario, pero primero debe parar antes de cambiar de giro.

2. Diseñar la unidad de control para implementar una máquina de venta de gaseosas usando el procesador de la Figura 1: RTL y ASM (20%):
   - recibe monedas de 100, 200, 500 pesos, valores almacenados en R10, R11 y R12
   - productos: coca-cola: 1000, pepsi: 800 y postobon 600 pesos, valores almacenados en R7, R8 y R9
   - la maquina entrega la devuelta

3. Diseñar un circuito controlador para realizar la multiplicación de dos datos que se encuentran en R5 y R6 y el resultado debe ser almacenado en R2 y R3. RTL y ASM (30%)
   ALU→ 00: Transferir A; 01: Sumar A + B; 10: Restar A – B; 11: Transferir B
   Shifter→ 00: *no shift* ; 01: *shift left,0* ; 10: *shift left,1*; 11: *shift right,0*





4. Diseñar un circuito controlador para realizar la raíz cuadrada de un dato que se encuentra en R5 y el resultado se almacena en R6 y R7. RTL y ASM (30%)

NOTA:
1. Los diagramas de estado solo tendrán tres notas: 5.0 si el diagrama de estados es bueno y óptimo (un número mínimo de estados); 2.5 si el diagrama es bueno pero no óptimo y 1.0 si el diagrama esta malo.

Universidad del Valle
Facultad de Ingeniería, Escuela EIEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2 horas

Juanita Rueda E. 5 #
0831135 - Enero 20 del 2012
#EpicFail

1. Diseñar un registro de desplazamiento de 8 bits que permita desplazar 1 o 2 bits en un solo ciclo de reloj. Diagrama de estados, usar FF-D (35%)
2. Diseñar la unidad de control para implementar una máquina de venta de gaseosas usando el procesador de la Figura 1. *RTL y ASM (15%)*
   - recibe monedas de 100, 200, 500 y 1000 pesos, estos valores son almacenados en la RAM
   - valor de productos: coca-cola= 1800; pepsi= 1400; postobon= 1200; Hit= 1000 pesos
   - la maquina es honrada y entrega la devuelta
3. Diseñar un circuito controlador para realizar la "*multiplicación*" usando el algoritmo de Booth, los datos se encuentran en R0 (multiplicando) y R1 (multiplicador), y el producto debe ser almacenado en R3 (parte alta) y R4 (parte baja). El algoritmo de Booth codifica el multiplicador así: 00 y 11 realiza la operación de desplazamiento, 01 realiza la operación de sumar el multiplicando, 01 realiza la operación de restar el multiplicando: Ejemplo: sea el multiplicador 1011*(0)*, la codificación es desde izquierda a derecha: 10→sumar el multiplicando, 11→ no-operación solo desplazar, 01→restar el multiplicando, 10→ sumar el multiplicando *RTL y ASM* (35%)

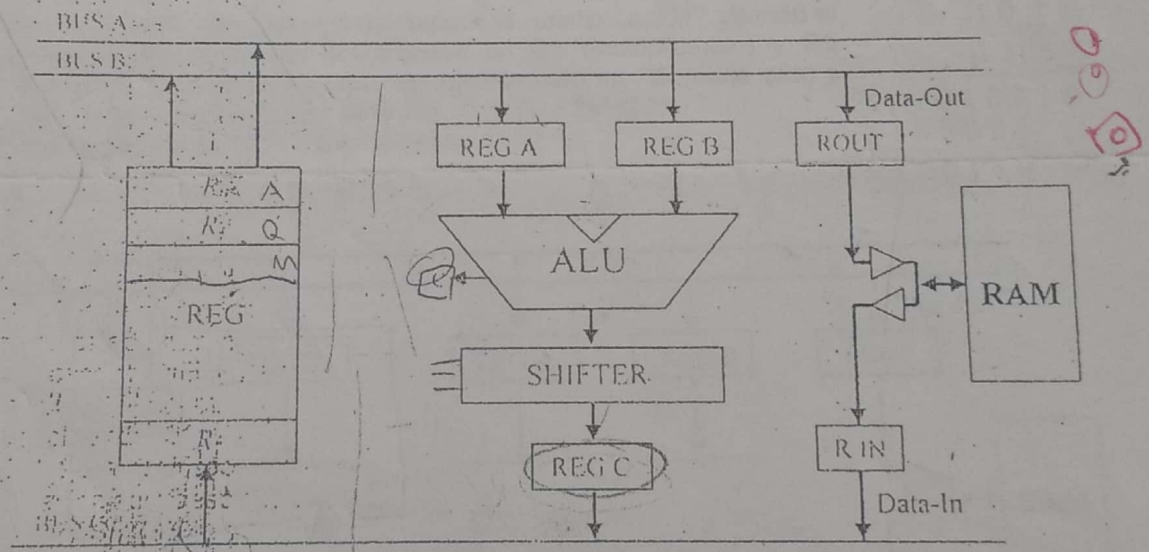


Figura 1: Procesador UV2012

4. Diseñar un circuito controlador para realizar la "*raíz cuadrada*" de un dato que se encuentra en el registro R2 y el resultado debe ser almacenado en R3: *RTL y ASM (15%)*

Nota:
ALU (S_1S_0): 00 = A or B ; 01= A and B ; 10 = A+B ; 11 = A-B
Shifter (S_1S_0): 00= lógico shift-left; 01= lógico shift-right; 10= aritmetico shift-left; 11= aritmetico shift-right.
Los diagramas de estado solo tendrán tres notas: 5.0 para bueno y óptimo (un número mínimo de estados); 2.5 para bueno pero no óptimo y 1.0 para malo.

$R_A \leftarrow R_0$, $R_B \leftarrow R_1$
$R_C \leftarrow R_A + R_B$
$R_C \leftarrow Shl(R_A + R_B, 0)$

Segundo parcial

Universidad del Valle
Facultad de Ingeniería, Escuela EIEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2:30 horas

Diciembre 9 del 2013

1. Diseñar un registro de desplazamiento de 6 bits que permita realizar desplazamiento lógico a la derecha o a la izquierda de 1, 2 o 3 bits en un solo ciclo de reloj. El registro debe permitir cargar datos de 6 bits en forma síncrona y debe tener un "*reset*" asincrono. Diagrama de estados o ASM (15%)

2. Diseñar un *data-path* específico y la unidad de control para realizar la raíz cuadrada. Data-path y *ASM (10%)* y prueba de escritorio: dato = 11010101 (5%)

3. Diseñar la unidad de control para realizar la operación $(X^{1/2})^{1/2}$. El dato X (6-bits) está en R4, la raíz debe ser almacenada en R5 y el residuo en R6 del procesador de la Fig. 1. *RTL-ASM (20%)*

4. Diseñar la unidad de control para realizar la operación X^4, el dato X (4-bits) está en R4, el resultado debe ser almacenada en R5,R6,R7,R8 del procesador de la Fig. 1. Usar el algoritmo de **Booth** para la multiplicación RTL-*ASM* (20%)

5. Diseñar un circuito controlador para realizar la *multiplicación"* usando el algoritmo de **Radix 4**, los datos están en R4 (multiplicando) y R5 (multiplicador), y el producto debe ser almacenado en R6 (parte alta) y R7 (parte baja) del procesador de la Fig. 1. RTL-*ASM* (30%)
   Ejemplo: multiplicador 1011 y multiplicando 1110:

```
  1 0 1 1
  1 1 1 0
---------
  0 1 0 1 1 0
1 0 0 0 0 1
-----------
1 0 0 1 1 0 1 0
```

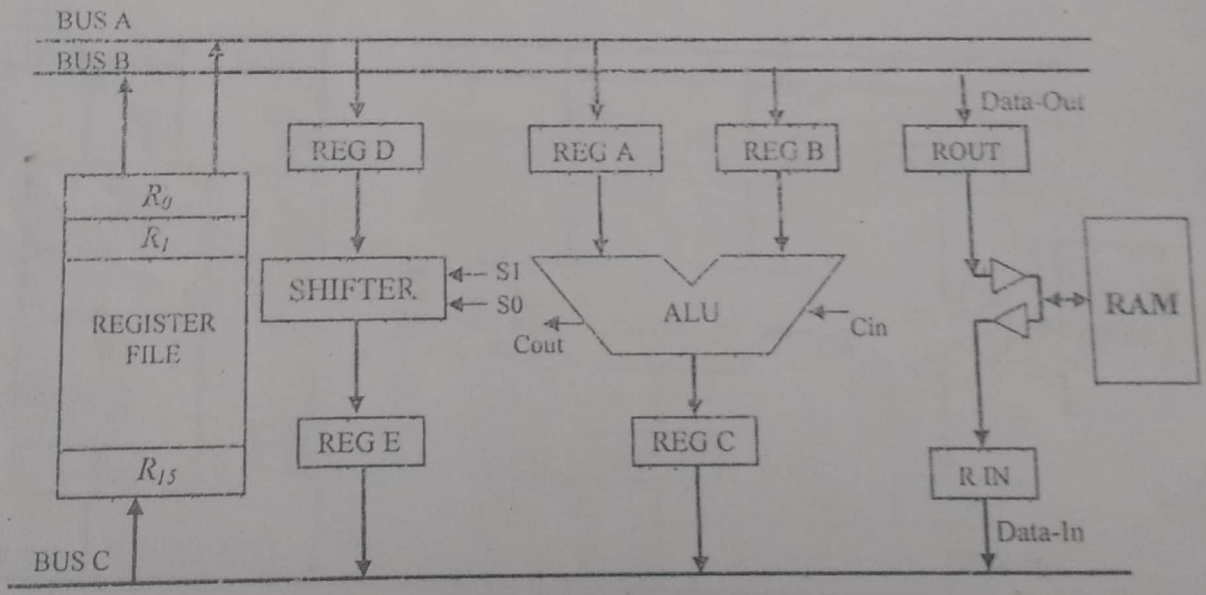


Figura 1: Procesador UV4

Nota:

- Shifter (S_1S_0): 00 = lógico-left {0} ; 01 = lógico-left {1} ; 10 = lógico-right {0} ; 11 = lógico-right{1}
- ALU (S1S0): 00 = A or B ; 01= A and B ; 10 = A+B ; 11 = A-B
- Los ASMs y/o diagrama de estados solo tendrán tres notas: 5.0 para bueno y óptimo (un número mínimo de estados); 3.5 para bueno pero no óptimo y 1.0 para malo.

Universidad del Valle
Facultad de Ingeniería, Escuela EIEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2:30 horas

Diciembre 14 del 2013

1. Diseñar una FSM Mealy para controlar un motor DC. Diagrama de estados (20%)
- El motor arranca y gira en sentido normal cuando detecta la secuencia 01101; salida $Z_1Z_0 = 01$
- El motor arranca y gira en sentido contrario cuando detecta la secuencia 00010; salida $Z_1Z_0 = 10$
- El motor se detiene cuando detecta la secuencia 1010; salida $Z_1Z_0 = 00$
- El motor gira en sentido contrario cuando detecta la secuencia 0101pero primero debe parar antes de cambiar de giro; salida $Z_1Z_0 = 00^*$.

2. Diseñar un *data-path* específico y la unidad de control para realizar *división*. Data-path y *ASM (10%)* y prueba de escritorio: dividendo = 11010101 y divisor 00010101(5%)

3. Diseñar la unidad de control para realizar la *operación 1/X*. El dato X (8-bits) está en R4, el resultado debe ser almacenado en R5 del procesador UV4. *ASM (20%)*

4. Diseñar un circuito controlador para realizar el *"logaritmo binario"* de un dato que se encuentra en R0 y el resultado debe ser almacenado en R1 y R2 del procesador UV4: *ASM* (20%).

5. Diseñar un circuito controlador para realizar la *"multiplicación"* usando el algoritmo de Radix 4, los datos están en R4 (multiplicando) y R5 (multiplicador), y el producto debe ser almacenado en R6 (parte alta) y R7 (parte baja) del procesador UV4. *ASM* (20%)
Ejemplo: multiplicador 1011 y multiplicando 1110

```
  1 0 1 1
  1 1 1 0
  ---------
  0 0 1 0 1 1 0
+ 1 0 0 0 1 1 0
  ---------
  1 0 0 1 1 0 1 0
```

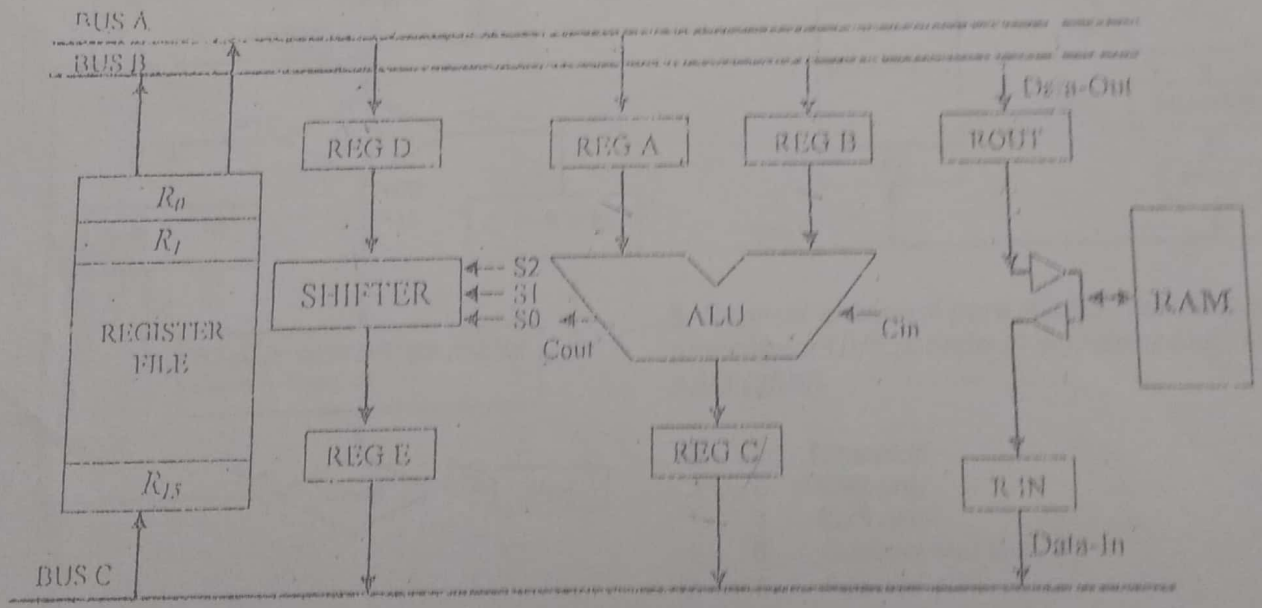


Figura 1: Procesador UV4

Nota:
- ALU (S_0S_1): 00 = A or B ; 01= A and B ; 10 = A+B ; 11 = A-B
- Shifter ($S_2S_1S_0$): 000 = lógico-left (0) ; 001 = lógico-left (1) ; 010 = lógico-right (0) ; 011 = lógico-right(1)
  1XX = transferir
- Los ASMs solo tendrán tres notas: 5.0 para bueno y óptimo (un número mínimo de estados); 2.5 para bueno pero no óptimo y 1.0 para malo.

Parcial 2

Universidad del Valle
Facultad de Ingeniería, Escuela IEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2 horas y 30 minutos

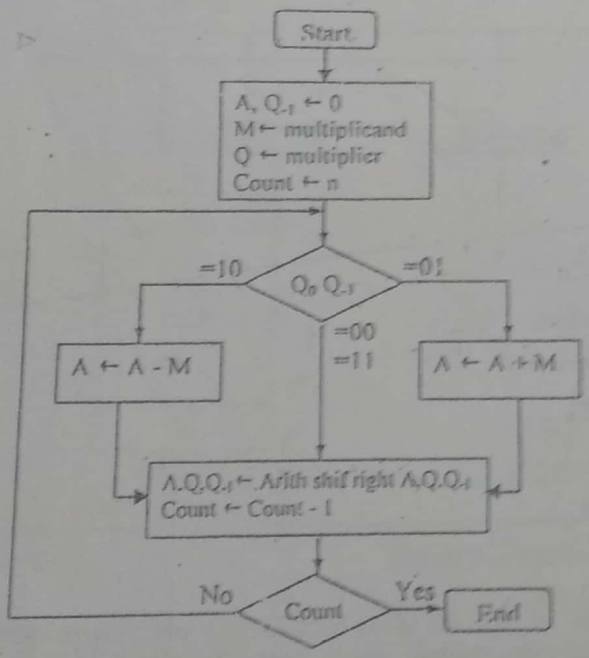
Junio 4 del 2014

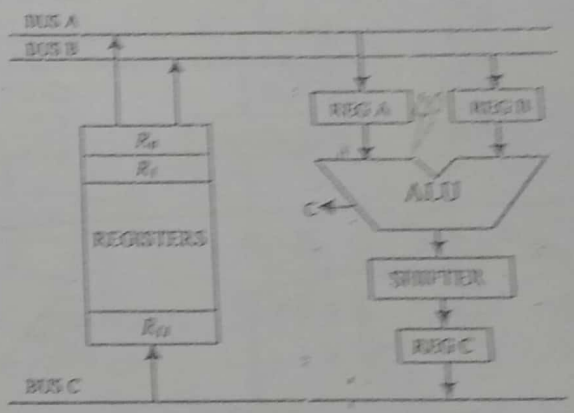
1. Diseñar un circuito detector de secuencias: (15%)
-Si detecta 00,11,10 la salida es 01 y permanece en 01 hasta detectar 10,00,10, que lleva a la salida a 00
-Si detecta 10,00,01 la salida es 10 y permanece en 10 hasta detectar 01,11,01, que lleva a la salida a 00
-Existe superposición, en este caso, si la salida está en 01 puede pasar a 10 si detecta la respectiva secuencia, y viceversa. La FSM debe ser Mealy

2. Diseñar una FSM Mealy síncrona para controlar un motor DC: (15%).
- El motor arranca y gira en sentido normal cuando detecta la secuencia 00110; salida Z1Z0 = 01
- El motor arranca y gira en sentido contrario cuando detecta la secuencia 11001; salida Z1Z0 = 10
- El motor se detiene cuando detecta la secuencia 1010; salida Z1Z0 = 00
- El motor gira en sentido contrario cuando detecta la secuencia 0101 pero primero debe parar antes de cambiar de giro; salida Z1Z0 = 00*.

3. Diseñar un data-path específico y la unidad de control para realizar división. Data-path y ASM (15%) y prueba de escritorio: dividendo = 11010101 y divisor 00010101(5%)

4. Diseñar la unidad de control para realizar la operación $1/X^2$. El dato X (8-bits) está en R1, el resultado debe ser almacenado en R2 del procesador UV3. ASM (25%)





5. Realizar el control para realizar la multiplicación en el procesador UV3 usando el algoritmo de Booth radix-2. ASM (20%)

Qi	Qi-1	Operation
0	0	Shift only
1	1	Shift only
1	0	Subtract and shift
0	1	Add and shift

Ejemplo: Multiplicador Q= 1010, codificación: 00; 10; 01; 10
Realizar la prueba de escritorio para el ejemplo: 8x(-9)=72 (5%)

Nota: ALU→ 00: Transferir A; 01: Sumar A + B; 10: Restar A – B; 11: Función A and B
Shifter→ 1XX: no shift ; 000: logical shift left, 0 ; 001: logical shift right,0 ; 010: arithmetic shift right ; 011: shift left,1
Los diagramas de estado y ASM solo tendrán tres notas: 5.0 (óptimo); 3.5 (bueno, pero no óptimo) y 1.0 (malo)

Universidad del Valle
Facultad de Ingeniería, Escuela EIEE
Curso Sistemas Digitales II: Ingeniería Electrónica
Profesor: Jaime Velasco Medina
Tiempo: 2:30 horas

Nov. 26 del 2014

1. Diseñar un *data-path* específico y la unidad de control para realizar la raíz cuarta. *ASM (10%)* y prueba de escritorio: dato = 111001 (5%)

2. Diseñar *data-path* específico y la unidad de control para calcular X^4. *ASM* (10%) y prueba de escritorio: dato = 1001 (5%)

3. Diseñar un circuito controlador para realizar la división N/N bits. El dividendo se encuentra en R1, y el divisor en R2 del banco de registros del procesador UV-ARM7 de la Figura 1. El cociente y el residuo deben ser almacenados en R6 y R7, respectivamente. *RTL-ASM* (15%) y prueba de escritorio: Dividendo = 1101 y divisor = 0011 (5%).

4. Diseñar un circuito controlador para realizar la multiplicación basada en el algoritmo de Booth. El multiplicando se encuentra en R1, y el multiplicador en R2 del banco de registros del procesador UV-ARM7 de la Figura 1. El producto debe ser almacenado en ~~R1~~ y ~~R2~~ R3 R4. *ASM* (20%) y prueba de escritorio: multiplicando = 1011 y multiplicador = 1011 (5%).

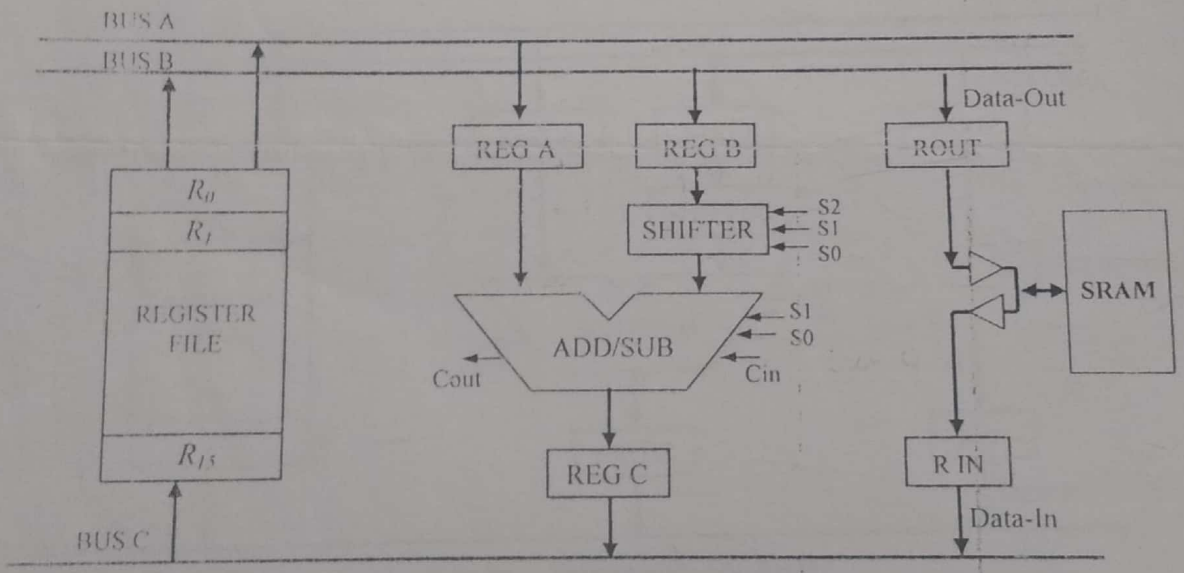


Figura 1: Procesador UV-ARM7

5 Diseñar un circuito controlador para realizar la "*multiplicación*" usando el algoritmo de **Radix 4**, los datos están en R3 (multiplicando) y R4 (multiplicador), y el producto debe ser almacenado en R5 (parte alta) y R6 (parte baja) del procesador UV-ARM7 de la Fig. 1. RTL-*ASM* (25%)
Ejemplo: multiplicador 1011 y multiplicando 1110.

```
  1 0 1 1
  1 1 1 0
-----------
  0 1 0 1 1 0
1 0 0 0 0 1
-----------
1 0 0 1 1 0 1 0
```

Nota:
- ALU (S_1S_0): 00 = A or B ; 01= A and B ; 10 = A+B ; 11 = A-B
- Shifter (S2S1S0): 000 = lógico-left {0} ; 001 = lógico-left {1} ; 010 = lógico-right {0} ; 011 = lógico-right{1}
- 1XX = transferir
- Los ASMs solo tendrán tres notas: 5.0 para bueno y óptimo (un número mínimo de estados); 2.5 para bueno pero no óptimo y 1.0 para diagrama malo.