Para programar en ARM

<https://salmanarif.bitbucket.io/visual/>

Para convertir el código ARM en ensamblador

<http://armconverter.com/>

<http://shell-storm.org/online/Online-Assembler-and-Disassembler/>

Propuesta:

1. Dinosaur jump chrome:v
2. Pokemon catch simulator

Para crear sprite se puede utilizar PISKEL para el diseño de las figuras.

Guardar en una paleta de colores, cada color y después pintarlos con un deco. Cambiar los hexa que se exportan a 0x to 24”h y ver en la paleta cada color para el deco. Ya que lo que se importan y son pocos colores se decodifican en una pequeña cantidad de bits y en el deco se pone el cambio.

Las dos propuestas que se tienen son:

1. Bichito Saltarin.
2. Atrapa al bichito.

Sprite de objetos en ensamblador.

<http://www.sromero.org/ext/articulos/gfx/graf6.html>

<https://asm86.wordpress.com/2011/03/11/tiles-y-sprites-en-ensamblador/>

Ejemplo de juego

<https://imanoleasgames.blogspot.com/search/label/Last%20Crown%20Warriors>

<https://www.retrogames.cl/cap1.php>

Controlodador Ps2

<https://www.instructables.com/id/PS2-Keyboard-for-FPGA/>

Listado componentes procesador

ALU

MUX 2:1

Registros, PC

Memoria de Instrucciones

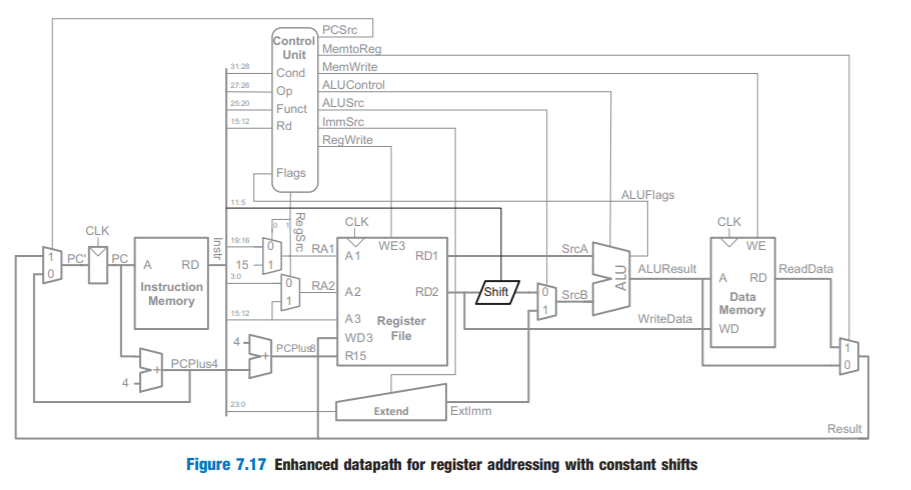
Memoria de Datos

Extensión

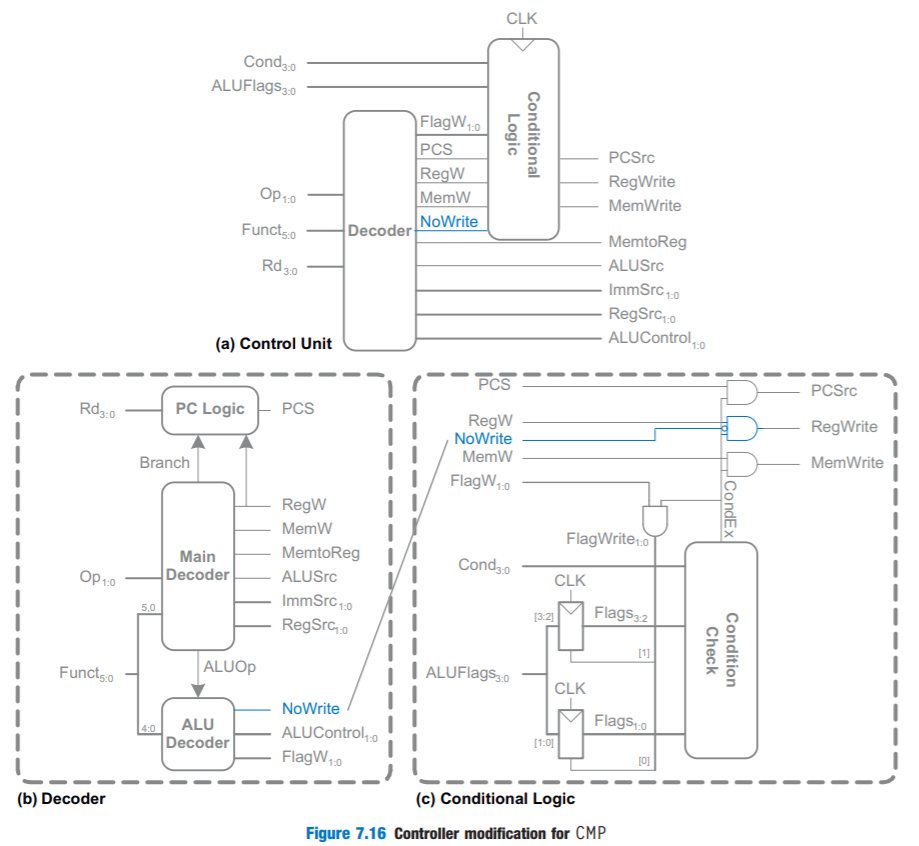
Banco de registros

Sumador (+4)

Datapath



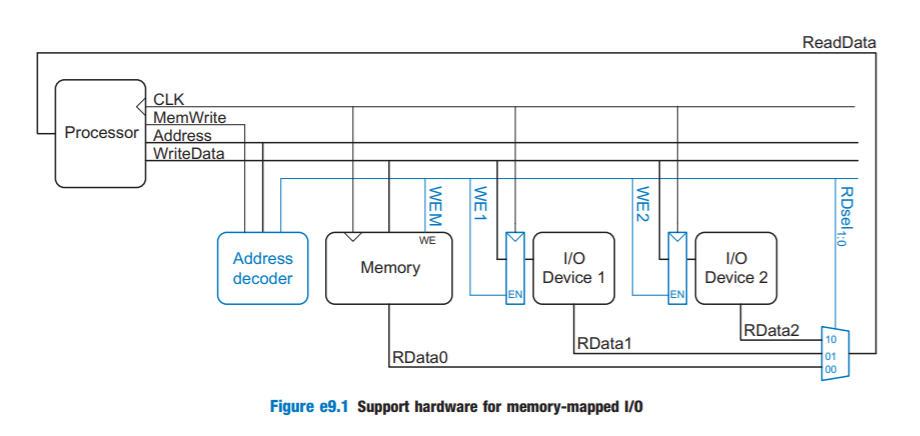
Control



CPU+Memory

.0

CPU + I/O



POSICIONES DE MEMORIA

32'h0FA0: //Sprite Bichito

32'h0FA4: //Sprite Obstáculo

32'h0FA8: //PS2

cmd Name Description Operation 0000 AND Rd, Rn, Src2 Bitwise AND Rd ← Rn & Src2 0001 EOR Rd, Rn, Src2 Bitwise XOR Rd ← Rn ^ Src2 0010 SUB Rd, Rn, Src2 Subtract Rd ← Rn – Src2 0011 RSB Rd, Rn, Src2 Reverse Subtract Rd ← Src2 – Rn 0100 ADD Rd, Rn, Src2 Add Rd ← Rn+Src2 0101 ADC Rd, Rn, Src2 Add with Carry Rd ← Rn+Src2+C 0110 SBC Rd, Rn, Src2 Subtract with Carry Rd ← Rn – Src2 – C 0111 RSC Rd, Rn, Src2 Reverse Sub w/ Carry Rd ← Src2 – Rn – C 1000 (S = 1) TST Rd, Rn, Src2 Test Set flags based on Rn & Src2 1001 (S = 1) TEQ Rd, Rn, Src2 Test Equivalence Set flags based on Rn ^ Src2 1010 (S = 1) CMP Rn, Src2 Compare Set flags based on Rn – Src2 1011 (S = 1) CMN Rn, Src2 Compare Negative Set flags based on Rn+Src2 1100 ORR Rd, Rn, Src2 Bitwise OR Rd ← Rn | Src2 1101 Shifts: I = 1 OR (instr11:4 = 0) MOV Rd, Src2 Move Rd ← Src2 I = 0 AND (sh = 00; instr11:4 ≠ 0) LSL Rd, Rm, Rs/shamt5 Logical Shift Left Rd ← Rm << Src2 I = 0 AND (sh = 01) LSR Rd, Rm, Rs/shamt5 Logical Shift Right Rd ← Rm >> Src2