**Temas básicos**

* Arq Von Neumann
* Representación de datos
* Logica, funciones y circuitos
* Instrucciones y maquinas
* ciclo de instruccion
* simulador

**Concepto de programa :**

Para cada paso se necesita un nuevo conjunto de señales de control.

Las instrucciones proporcionan esas señales de control

Aparece el nuevo concepto de programación

¡No hay que cambiar el hardware!

**Arq von Neumann :**

La unidad central de procesamiento(cpu) está constituida por la unidad de control (uc) y la unidad arit-log(alu)

datos e instrucciones deben introducirse en el sist y los resultados se proporcionaran mediante componentes de e/s--> la maq genera resultados

se necesita almacenar temporalmente datos e instrucciones: mem principal

**Tipos de datos :**

Las Pc manejan 4 tipos basicos de datos binarios:

n enteros con/sin signo

n reales con signo

n decimales codificados en binario (BCD)

caracteres

**representacion de numeros enteros :**

sin signo (bss)

con signo

Modulo y y signo (BCS)

ca1-> complemento a la base reducida

ca2-> complemento a la base

exceso

**Rango y resolucion :**

rango: dif entre el num mayor y el num menor

resolucion: dif entre 2 num consecutivos

**Num punto flotante :**

-un num se puede representar de la forma + - Mx B^(+E)

-este num se puede almacenar en una palabra binaria con 2 campos: mantisa m y exponente e

-m y e estan representados en algunos de los sist en pto fijo como bss. bcs, ca2, ca1, exceso

**error absoluto y relativo :**

EA: es la dif entre el valor representado y el valor a representar

error abs: max <= resolucion/2

error relativo: ea/ num a representar

**El nivel de logica digital :**

- un circuito digital es en el que estan presentes dos valores logicos

- compuertas logicas son dispositivos electronicos que pueden realizar distintas funciones logicas

con los valores logicos en su entrada generando un valor logicos de salida

- las compuertas basicas son: AND, OR, NOT, NAND, NOR Y XOR

F = A.B' + A'.B = XOR

**Repertorio de instrucciones :**

- Es el conjunto completo de instrucciones que se realizan en una CPU

-codigo maquina

-binario

- Representado simbolicamente por un conjunto de codigos de ensamblaje

-de operaciones:

-ADD, SUB, LOAD (cargar datos en un registro)

-de operandos:

-ADD BX, PEPE; (sumar contenidos de reg BX y direcciones PEPE, el resultado se guarda en reg BX)

**ELEMENTOS DE UNA INSTRUCCION:**

-- Codigo de operacion (cod op)

--Referencia a operandos fuentes

--Referencia al operando resultado

--Referencias a la siguiente instrucción

**TIPOS DE INSTRUCCIONES :**

--Procesamiento de datos:

-Instrucciones aritmeticos-logicas

--Almacenamiento de datos:

-Instrucciones de memoria

--Transferencia de datos:

-Instrucciones de E/S

--Control:

-Instrucciones de testeo y flujo del programa

**DESICIONES EN EL DISEÑO DEL CONJUNTO DE INSTRUCCIONES:**

--Tipos de operando (datos)

-datos enteros, punto flotante, coma fija (entere y fraccionaria), caracteres, codigo ascii)

--Repertorio de operaciones

-Cuantas operaciones se considerará ? - suma, resta, mul, div

-Cuales operaciones se realizaran ?

-Cuan compleja sera cada una de ellas ?

--Formatos de instrucciones:

-Longitud de instruccion

-Numero de direcciones

-Tamaño de los campos

--Registros

-numero de registros de la CPU referenciables

-En que registros se pueden ejecutar que operaciones?

--Modos de direccionamiento

-como es especificada la ubicacion de un operando o una instruccion ?

sirven para indicar a la unidad de control donde estan cada uno de los operandos que quiero realizar, y cuando genera el resultado

decir donde ira ubicado

**TIPOS DE OPERANDO :**

--Direcciones

--Numeros

-pfijo o pflotante

--Caracteres

-ascii, ebcdic. etc

--Datos logicos

Bits (1 o 0) ej: flags o indicadores

**TIPOS DE OPERACIONES :**

--Transferencia de datos

-Debe especificarse:

-ubicacion del operando fuente

-ubicacion del operando destino

-tamaño de los datos a ser transferidos

-modos de direccionamiento

-Diferentes movimientos -> diferentes instrucciones

-reg,reg ; reg,mem ; o mem,reg

- O una instruccion y diferentes direcciones

-MOV destino, fuente ; copia fuente a destino

--Aritmeticas

- add, sub, mul, div (utiliza la memoria y la ALU )

- numeros enteros sin/con signo

- numeros en punto flotante ?

-Pueden incluirse operaciones ....

-Inc o dec (en 1 el operando)

-Neg: cambia el signo del operando (Ca2) (necesita la unidad de control ademas de la ALU)

-Abs: toma el valor absoluto del operando

-shift left/right: desplaza bits a izq/der un lugar

--Logicas

Operaciones que manipulan bits individualmente

-operaciones booleanas

AND, OR, XOR, NOT

Otras operaciones

-Rotate left/right: rota las posiciones de los bits a izq/der

--Conversion

Operaciones para cambiar formato de datos

-Conversion de binario a decimal o de EBCDIC A ASCII

--E/S

-Pocas instrucciones pero de acciones especificas

-IN o OUT

-Se pueden realizar utilizando inistrucciones de movimiento de datos

-MOVE

-Se pueden realizar a traves de un controlador aparte: DMA (Direct Memory Access)

--Control del sistema

--Control de flujo (SALTO) (en estas instrucciones necesito el Contador de Programa y la ALU)

-Modifican el valor contenido en el registro PC

-Salto Incondicional

-JMP equis ; saltar a la posicion 'equis'

-Salto Condicional

-JZ equis ; saltar a la posicion equis, si bandera Z = 1 (mdd propio, la direccion queda almacenada ahi)

-Salto con retorno o llamada a subrutina

-CALL subrut ; saltar a la posicion 'subrut' (automaticamente cuando se llama guarda la direccion de retorno)

Para retornar al programa que llamó, se debe utilizar la instruccion RET como ultima del cuerpo de subrutina

-- FORMATO DE INSTRUCCION

maquina de 4 direcciones:

COD OP - dirOP1 - dirOP2 - dirRES - dirProxInstr

A = [(B+C)\*D] / E

PROGRAMA

I0 ADD dirB, dirC, dirA, dirI3

I1 HALT

I2 DIV dirA, dirE, dirA, dirI1

I3 MUL dirA, dirD, dirA, dirI2

maquina de 3 direcciones

COD OP dirOP1 dirOP2 dirRES

A = [(B+C)\*D] / E

PROGRAMA CPU con registro PC

I0 ADD dirB, dirC, dirA

I1 MUL dirA, dirD, dirA

I2 DIV dirA, dirE, dirA

I3 HALT

maquina de 2 direcciones

COD OP - dirOP1/dirRES - dirOP2

A = [(B+C)\*D] / E

PROGRAMA CPU con registro PC y nuevas operaciones

I0 MOV dirA, dirB

I1 ADD dirA, dirC

I2 MUL dirA, dirD

I3 DIV dirA, dirE

I4 HALT (fin de instruccion)

maquina de 1 direcciones

COD OP - dirOP2

A = [(B+C)\*D] / E

PROGRAMA CPU con registro PC y registro acumulador y nuevas operaciones

I0 LOAD dirB

I1 ADD dirC

I2 MUL dirD

I3 DIV dirE

I4 STORE dirA

I5 HALT (fin de instruccion)

maquina de 0 direcciones

COD OP - dirOP2

A = [(B+C)\*D] / E

PROGRAMA CPU con registro PC, PILA(adentro de la cpu) y nuevas operaciones

I0 PUSH dirE

I1 PUSH dirD

I2 PUSH dirC

I3 PUSH dirB

I4 ADD

I5 MUL

I6 DIV

I7 POP dirA

I8 HALT

--los saltos condicionales usan indirecto por desplazamiento

-basado(ESTA PENSADO PARA DAR VELOCIDAD DE EJECUCION)

-indexado(el registro se modifica el valor cada vez que se usa) - para recorrer vectores

-relativo al PC. -- indirecto via registro contador de programa (SALTOS CONDICIONALES)

y en un salto incondicional, se puede usar el directo de memoria, o directo de registro o indirecto por registro

**MODOS DE DIRECCIONAMIENTO :**

-Inmediato

-Directo de memoria o Absoluto

-Directo de Registro

-Indirecto de memoria(en desuso)

-Indirecto con registro

-Indirecto con Desplazamiento

-basado(ESTA PENSADO PARA DAR VELOCIDAD DE EJECUCION)

-indexado(el registro se modifica el valor cada vez que se usa) - para recorrer vectores - se automodifica

-relativo al PC. -

-Pila (o relativo al SP) - se automodifica para apilar o desapilar un dato del programa

**Ciclo de instruccion básico :**

- Dos pasos:

-Captacion

-ejecucion

INICIO -->> CAPTAR LA SIG INSTRUCCION --> EJECUTAR LA INSTRUCCION --> PARADA

------------------------------------->

CICLO DE CAPTACION CICLO DE EJECUCION

**DIAGRAMA DE ESTADOS DEL CICLO DE INSTRUCCIÓN :**

calculo de la direccion de instruccion

captacion de instruccion

decodificacion de la operacion de la instruccion

calculo de la direccion de operando

captacion de operando

operacion de datos

calculo de la direccion de operando

almacenamiento del operando

**Direccionamiento 2D(memoria estatica porque se hace con flip flops) :**

Este tipo de memoria posee un tamaño N. Debe tener una conexion que le permita leer/escribir datos y otra que indique la celda a escribir. Cada vez que se lee

o escribe, se lo hace a una celda completa.

Memorias de poco tamaño, lo cual es dificil de implementar con registros grandes(tamaño entre 512kb - 1mb)

Este tipop de direccionamiento se le aplica a memorias estaticas (cache y reg. del cpu) ya que se implementa con circuitos biestables(flip flops)

lo cual no necesitan refresco y son mas rapidas.

si tengo mil celdas y quiero poner 2mil, que hago ?

- replico las celdas pero a la primeras le agrego un bit 0 a cada celda y a la segundas les pongo un 1 para indicar que son distintas

x <------- 10 bits y se le suma un bit mas (11) para poder agregar un bit para especificar que memoria es cada una

'0' 0 00...0

0 00...1

0 1....0

.

.

.

'1023' 0 111..1

'1024' 1 00...0

1 00...1

1 01...0

1 10...0

.

.

.

'2047' 1 111..1

**direccionamiento 2D 1/2 (memoria dinamica porque se hace con transitores) :**

Memoria compuesta de varios chips. Se le provee al chip la direccion de memoria a acceder, la cual se divide en 2 partes: mediantes condensadores:

1 Indica a la memoria que "renglon" quiere leer

2 Buscar el bit correspondiente

Una vez identificado cada bit de la palabra, se reconstruye y se la ubica en una interfaz de salida.

A diferencia del direccionamiento 2D, que a pesar que tiene mas procesamiento para obtener una palabra, posee mas capacidad, son mas economicas y lentas. y necesitan

refresco constantemente.

**Localidad de referencia**: principio de localidad

Nos permite predecir que instrucciones y datos se utilizan en un futuro proximo

los casos mas importantes son 2:

localidad temporal: si una posicion de memoria es referenciada, es muy probable que la misma ubicacion sea referenciada en un futuro cercano

localidad espacial: si una localizacion de memoria es referenciada, es muy probable que las localizaciones cercanas a ella sean referenciadas

beneficio: el saber que yo en mi ciclo de instruccion tengo que ir primero a buscar la intruccion

el saber que por anticipado voy a tener la siguiente me permitiria armar distinto tipos de memorias o utilizar distintos tipos de memorias

pasamos de usar memoria lenta a registros mas rapidos

principio de localidad nos dara sustento a poder armar jerarquia de memoria

**JERARQUIA DE MEMORIA:**

--memoria del computador:

-tecnologias diferentes

-fundamentos fisicos distintos

-localizacion en lugares distintos

--objetivo :

-capacidad de almacenamiento

-tiempo de acceso reducido

TIPOS DE MEMORIA TIEMPO DE ACCESO TAMAÑO TIPICO

Registros 1 ns 1kb

caché 5-20 ns 1mb

mem. principal 60-80 ns 1gb

discos 10 milis 1 tb

1024\*1024 \* truecolor (3) \* (imagen) en un segundo