

PONTIFICIA UNIVERSIDAD JAVERIANA CALI

PROYECTO SISTEMA DE COMPUTO

INGENIERIA DE SISTEMAS Y COMPUTACION

FACULTAD DE INGENIERIA

MARIBELL SACANAMBOY

ARQUITECTURA DE COMPUTADOR II

GERMÁN ANDRÉS CAYCEDO MUTIS

DAVID HERNÁNDEZ CÁRDENAS

MARÍA CAMILA LÓPEZ LEAL

ALEJANDRO MEZA BARRERA

SANTIAGO DE CALI, 17 DE AGOSTO DE 2018

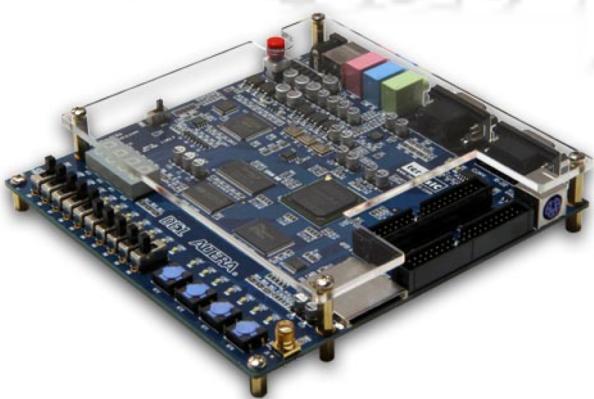
APLICACIÓN

Se desarrollará un sistema de cómputo donde se correrá una aplicación enfocada en el manejo, organización y conocimiento del periodo menstrual de las mujeres. Para ello se hará uso de información personal acerca de cada mujer, como la duración de su ciclo, la última vez de su periodo, la duración de días con periodo, entre otros. Además, la aplicación logrará alertar por medio de datos como la planificación y el hecho de estar en el periodo o no, cuándo se tienen señales de embarazo, retrasos anormales o en su defecto si su ciclo se encuentra correcto sin ninguna in tranquilidad. Al contrario de las aplicaciones normalmente vistas en las tiendas, que son abordadas para llevar un calendario y cómo con el tiempo se lleva una regulación del periodo menstrual, la nuestra se centrará en generar diferentes resultados inmediatos a partir de información predeterminada de cada mujer según sea su caso.



RECURSOS

La iniciación del sistema de cómputo se basa en el proceso de creación del algoritmo en alto y bajo nivel el cual será usada en el sistema ya mencionado. Para ello, se usarán aplicaciones de modificación de texto y comprobación del funcionamiento aritmético lógico de ambos algoritmos (bajo y alto nivel). Consumado esto, para la realización y/o el desarrollo del sistema de cómputo se hará uso de la herramienta Quartus II Edición Web para el análisis y el compendio de los diseños a realizar en lenguaje HDL (Lenguaje de descripción de hardware). En este caso, el lenguaje a emplear es VHDL derivado del anterior mencionado. El dispositivo en el cual se trabajarán las entradas, señales y salidas será la FPGA EP2C20F484C7.



ALTERA®



ALGORITMIA

El desarrollo de algoritmos en bajo nivel (.asm) dependen en gran parte de la invención de estos en alto nivel, que son los comúnmente realizados en el ámbito laboral y académico. Por lo tanto, la creación del programa a ejecutar por el procesador en lenguaje ensamblador se dio inicio con la creación del programa en lenguaje de alto nivel. En este caso, se hizo uso del lenguaje de programación C. Por lo tanto, la creación del algoritmo en bajo nivel se vio facilitada por el programa ya creado en alto nivel. Durante el proceso de desarrollo se tomó la posibilidad de asignarle una función específica dentro del programa a cada salida de este por medio de las diferentes entradas antes mencionadas (APLICACIÓN).

DIAGRAMAS DE FLUJO

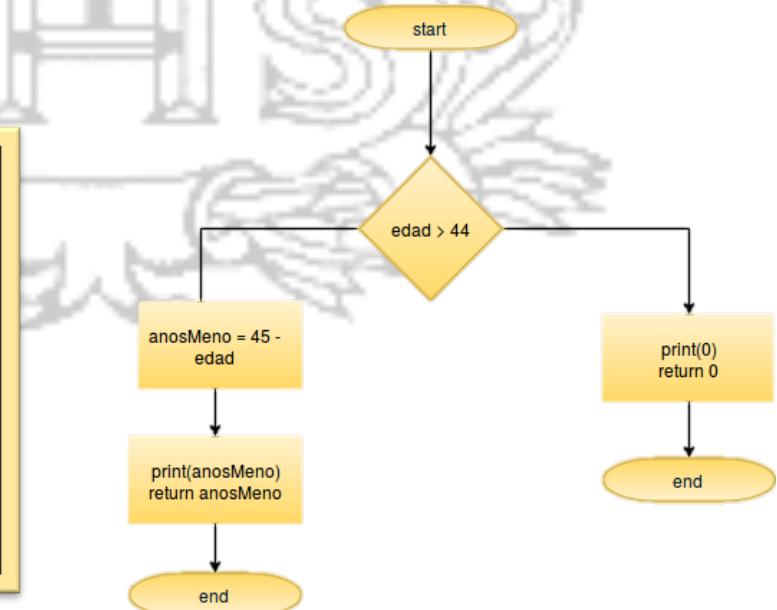
Los diagramas de flujo son bosquejos que describen sistemas, procesos y/o algoritmos en las ciencias de la computación. Su creación permite entender de manera sencilla y didáctica el funcionamiento de un programa sin un conocimiento amplio en informática o algoritmia. De este modo, se crearon diagramas de flujo para el programa en medio y bajo nivel (programa a ejecutar por el procesador). Al igual que en los códigos enunciados, los diagramas de flujo también están divididos según salidas específica

- **Medio Nivel:**

Código por bloques de funciones.

Menopausia:

```
int Menopausia (edad){  
    int anosmeno;  
    if (edad > 44){  
        printf("%d\n",0);  
        return 0;  
    }else{  
        anosmeno = 45-edad;  
    }  
    printf("%d\n", anosmeno);  
    return anosmeno;  
}
```

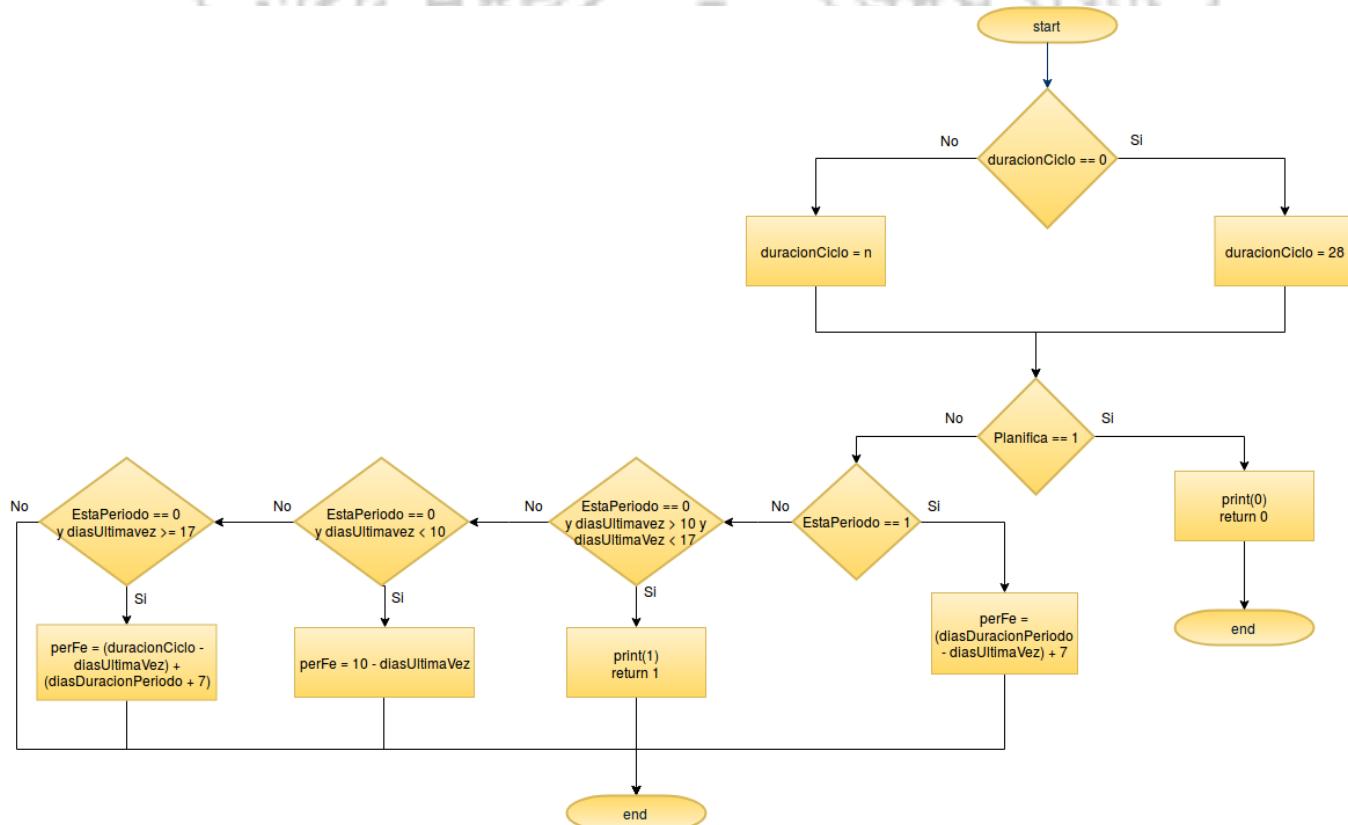


Periodo Fértil:

```

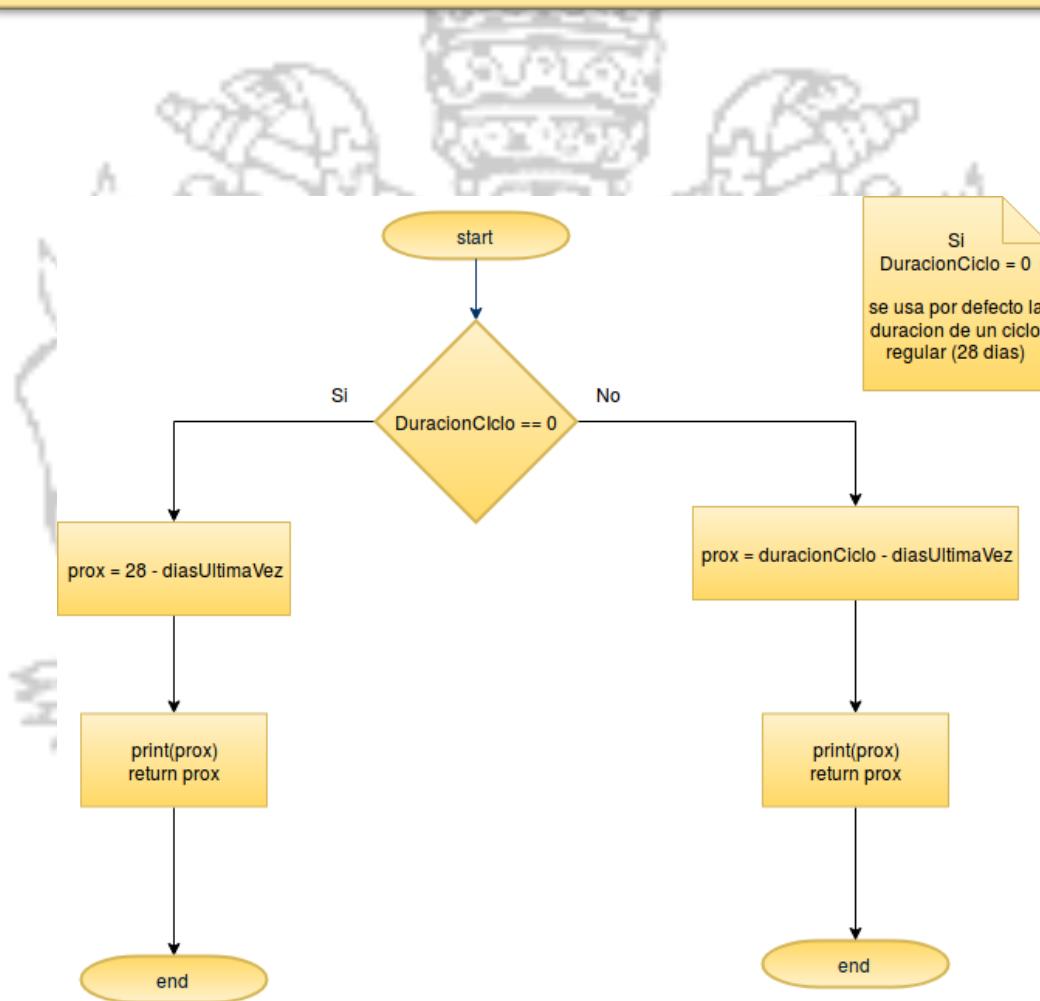
int PeriodoFertil(diasDuracionPeriodo, diasUltimaVez, duracionCiclo, Estaperiodo, planifica){
    //Función para conocer el dia del proximo periodo fertil
    int perFe;
    if (duracionCiclo == 0){
        duracionCiclo = 28;
    }
    if (planifica == 1){
        printf("%d\n",0);
        return 0; //No tiene periodo fertil.
    }else if(Estaperiodo == 1){
        perFe = (diasDuracionPeriodo - diasUltimaVez) + 7;
    }
    else if (Estaperiodo == 0 && diasUltimaVez > 10 && diasUltimaVez < 17){
        printf("%d\n",1);
        return 1; //Está en el periodo fertil.
    }
    else if(Estaperiodo == 0 && diasUltimaVez < 10){
        perFe = 10 - diasUltimaVez;
    }
    else if (Estaperiodo == 0 && diasUltimaVez >= 17){
        perFe = (duracionCiclo - diasUltimaVez) + (diasDuracionPeriodo + 7) ;
    }
    printf("%d\n",perFe );
    return perFe;
}

```



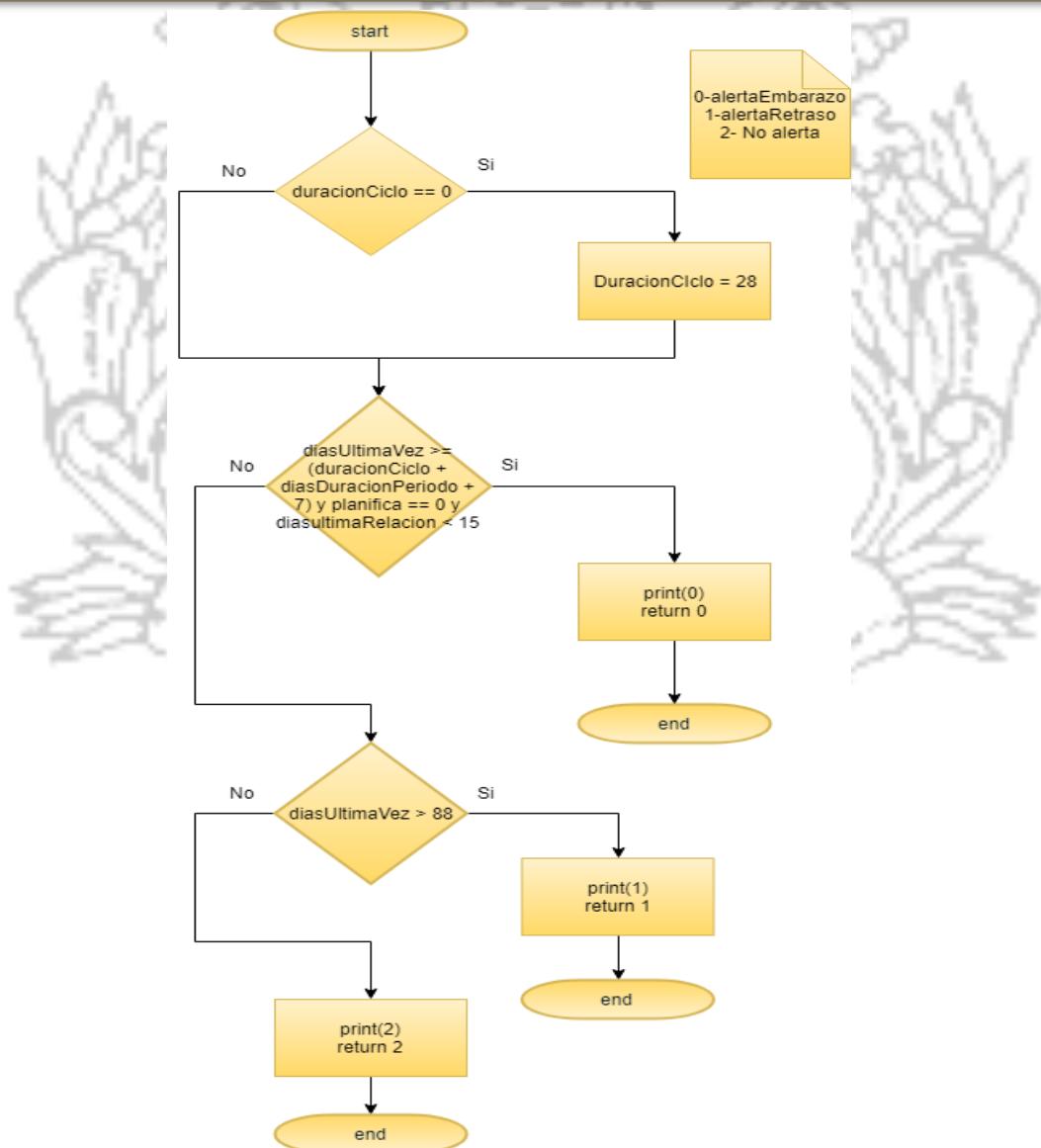
Próximo Periodo:

```
int ProximoPeriodo(diasUltimaVez, duracionCiclo){  
    //Función para conocer el dia del proximo periodo de menstruación  
    int prox;  
    if (duracionCiclo == 0){  
        prox = 28 - diasUltimaVez;  
    }  
    else{  
        prox = duracionCiclo - diasUltimaVez;  
    }  
    printf("%d\n",prox );  
    return prox;  
}
```



Alertas:

```
int Alertas(Estaperiodo, diasUltimaVez, duracionCiclo, planifica, diasultimaRelacion, diasDuracionPeriodo){  
    // 0 - alertaembarazo;  
    // 1 - alertatraso;  
    // 2 - no alerta;  
    if (duracionCiclo == 0){  
        duracionCiclo = 28;  
    }  
    if (diasUltimaVez >= (duracionCiclo + diasDuracionPeriodo + 7) && planifica == 0 && diasultimaRelacion < 15){  
        printf("%d\n",0);  
        return 0;  
    }  
    if (diasUltimaVez > 88){  
        printf("%d\n",1);  
        return 1;  
    }  
    printf("%d\n",2);  
    return 2;  
}
```



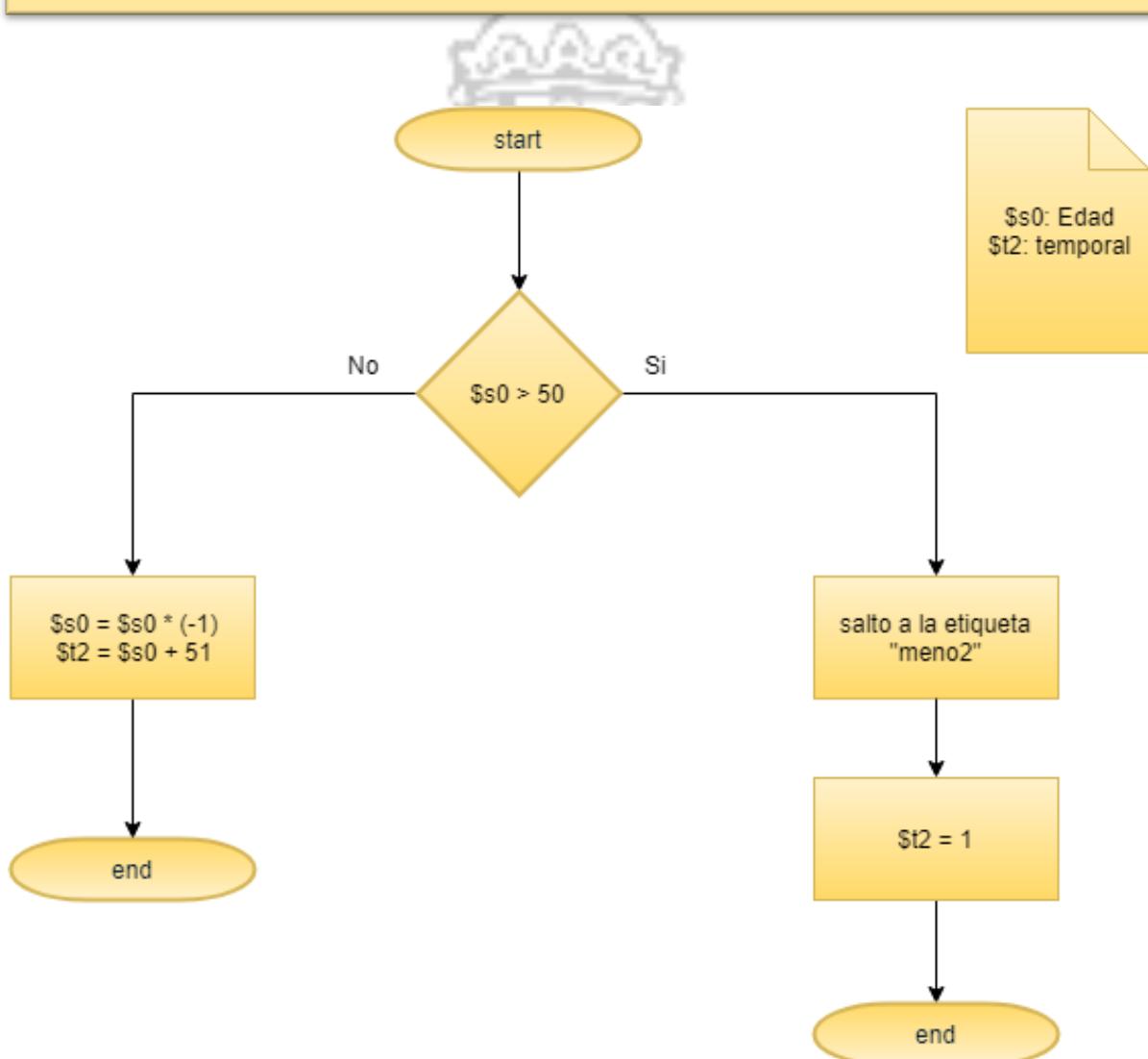
- **Bajo Nivel:**

Menopausia:

```

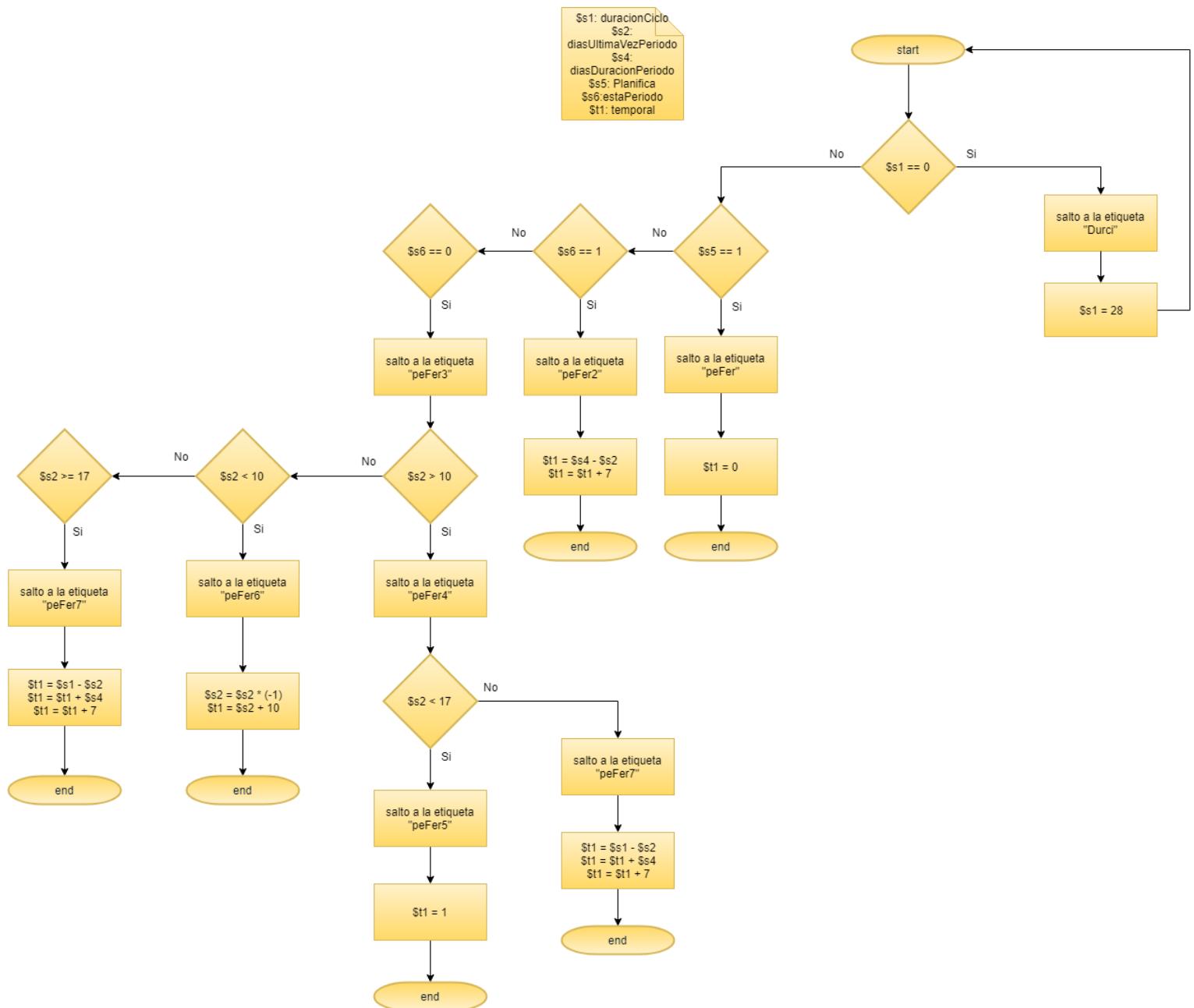
Menopausia:
    bgt $s0,50,meno2 #Si la edad es mayor a 50 salta a meno2
    mul $s0,$s0,-1
    addi $t2,$s0,51 #Años faltantes para tener menopausia
    j Alertas
meno2:
    addi $t2,$0,1 #Se le asigna uno porque esta prox a la menopausia

```



Periodo Fértil:

```
Prox:  
    mul $s2,$s2,-1  
    addi $t0,$s2,28 #Proximo periodo = 28 - diasultimavezperiodo  
  
PeriodoFertil:  
    blez $s1,DurCi #Si duracion ciclo es igual a 0 salta a Durci  
    beq $s5,1,PeFer #Si planifica es 1 salta a PeFer  
    beq $s6,1,PeFer2 # Si estaperiodo es 1 salta a PeFer2  
    beq $s6,0,PeFer3 #Si estaPeriodo es igual a 0  
  
PeFer3:  
    bgt $s2,10,PeFer4 #Si s6 = 0 y diasUltimaVez > 10  
    blt $s2,10,PeFer6 #Si S6 = 0 y diasUltimaVez < 10  
    bge $s2,17,PeFer7 #Si S6 = 0 y diasUltimaVez >= 17  
PeFer4:  
    blt $s2,17,PeFer5 #Si S6 = 0 y S2 > 10 y S2 < 17  
    j PeFer7  
PeFer5:  
    addi $t1,$t1,1 #Esta en su periodo fertil  
    j Menopausia  
PeFer6:  
    mul $s2,$s2,-1  
    addi $t1,$s2,10 #faltan t1 dias para su periodo fertil  
    j Menopausia  
PeFer7:  
    sub $t1,$s1,$s2 #Duracionciclo-DiasUltimaVez  
    add $t1,$t1,$s4 #(Duracionciclo-DiasUltimaVez)+diasDuracionPeriodo  
    addi $t1,$t1,7 #((S1+S2)+S4)+7  
    j Menopausia  
  
PeFer2:  
    sub $t1,$s4,$s2 # Perfe = diasduracionperiodo - diasultimavez  
    addi $t1,$t1,7 # Perfe = Perfe + 7  
    j Menopausia  
PeFer:  
    addi $t1,$0,0 #No tendrá periodo fertil  
    j Menopausia  
  
DurCi:  
    addi $s1,$0,28 #Duracion ciclo tendrá un promedio de 28 dias  
    j PeriodoFertil
```

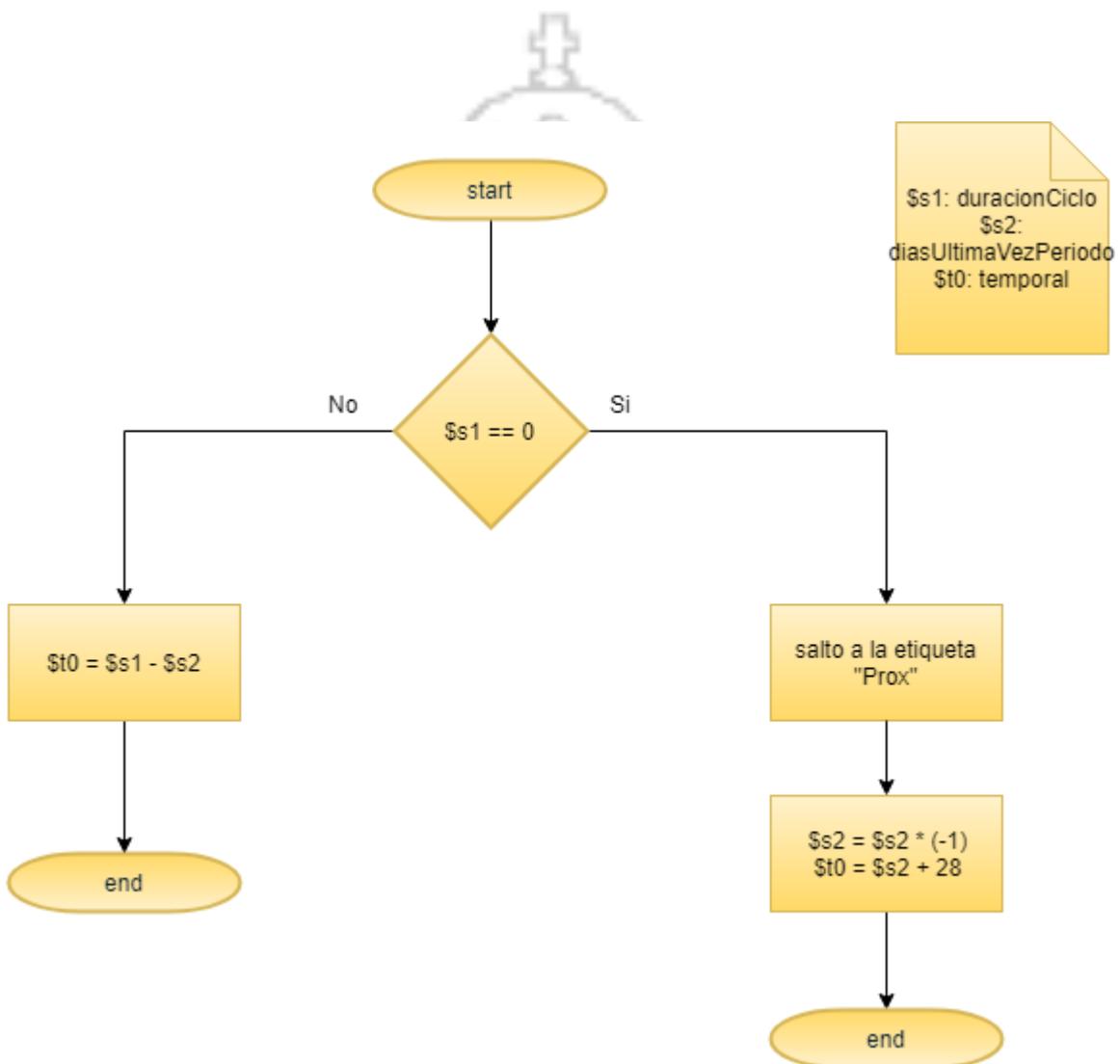


Próximo Periodo:

```

ProximoPeriodo:
    beq $s1,0,Prox #Si la duracionciclo es igual a 0 salta a Prox
    sub $t0, $s1,$s2 #De lo contrario proxperiodo = duracionciclo - diasultimavezperiodo
    j PeriodoFertil
Prox:
    mul $s2,$s2,-1
    addi $t0,$s2,28 #Proximo periodo = 28 - diasultimavezperiodo

```



Alertas:

```

Alertas:
blez $s1,Alerta2 #Si duracion ciclo es igual a 0 salta a Alerta2
add $t3,$s1,$s4 #Suma duracionCiclo + diasDuracionPeriodo de la condicion
addi $t3,$t3,7 #Suma la suma anterior mas 7 de la condicion
bge $s2,$t3,Alerta3 #Compara si diasUltimaVez >= (duracionCiclo + diasDuracionPeriodo + 7) se cumple y salta a Alerta3
j Alerta6

Alerta3:
beq $s5,0,Alerta4 #Si no planifica salta a Alerta5 a revisar la ultima condicion
j Alerta6

Alerta4:
blt $s3,15,Alerta5 #Si los dias de la ultima relacion son menores que 15 salta a Alerta5 porque se cumple la ultima condicion
j Alerta6

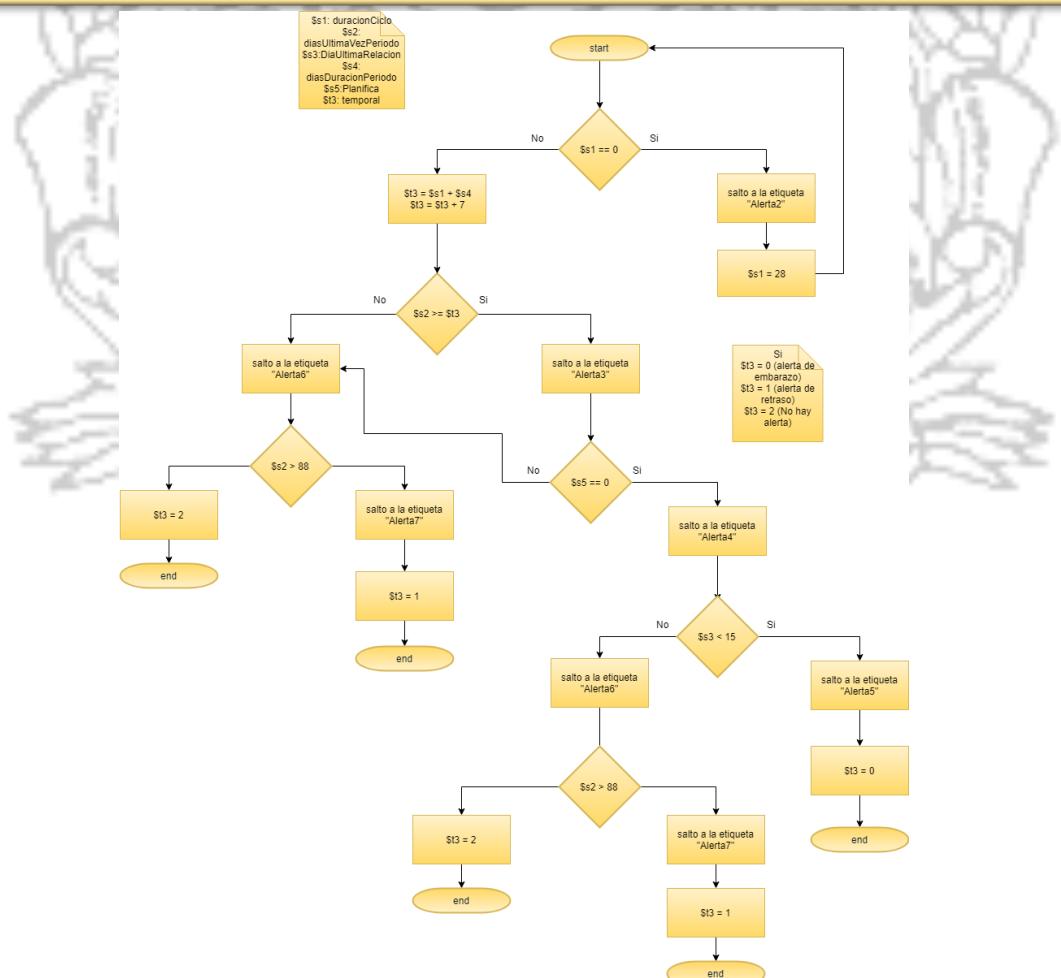
Alerta2:
addi $s1,$0,28 #Duracion ciclo tendra un promedio de 28 dias
j Alertas

Alerta5:
addi $t3,$0,0 #0 debido a que activa alerta de embarazo
j Fin

Alerta6:
bgt $s2,88, Alerta7
addi $t3,$0,2 #2 debido a que no activa ninguna alerta
j Fin

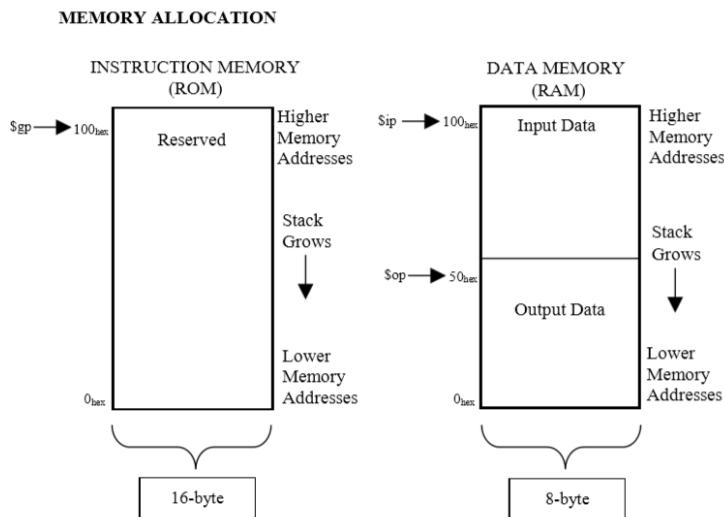
Alerta7:
addi $t3,$0,1 #1 debido a que activa alerta de retraso

```



CRITERIO DE DISEÑO PARA EL MANEJO DE MEMORIA

Para el manejo de la memoria dentro de la arquitectura se implementarán y/o usarán dos memorias (ROM y RAM). Esto facilitará la manera en que se acceden a las instrucciones y a los datos al momento de ejecutar el programa.



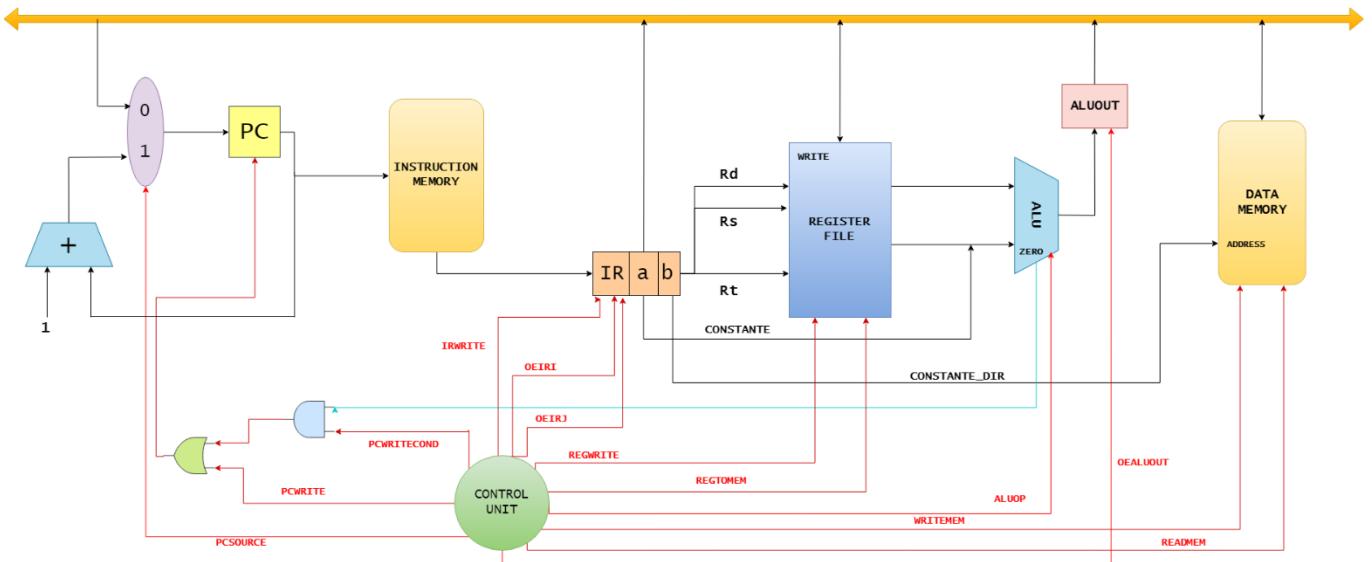
Por medio de nuevas instrucciones encargadas de generar input y output en la memoria de datos, los punteros se actualizarán permitiendo la llegada a los datos y, por ende, su obtención o almacenamiento de manera rápida.

```
//CARGAR DATO DE MEMORIA A REGISTRO
Output:
    beq sp,dirección,Output1
    addi sp,sp,#1
    j output
Output1:
    addi registro,(dirección)

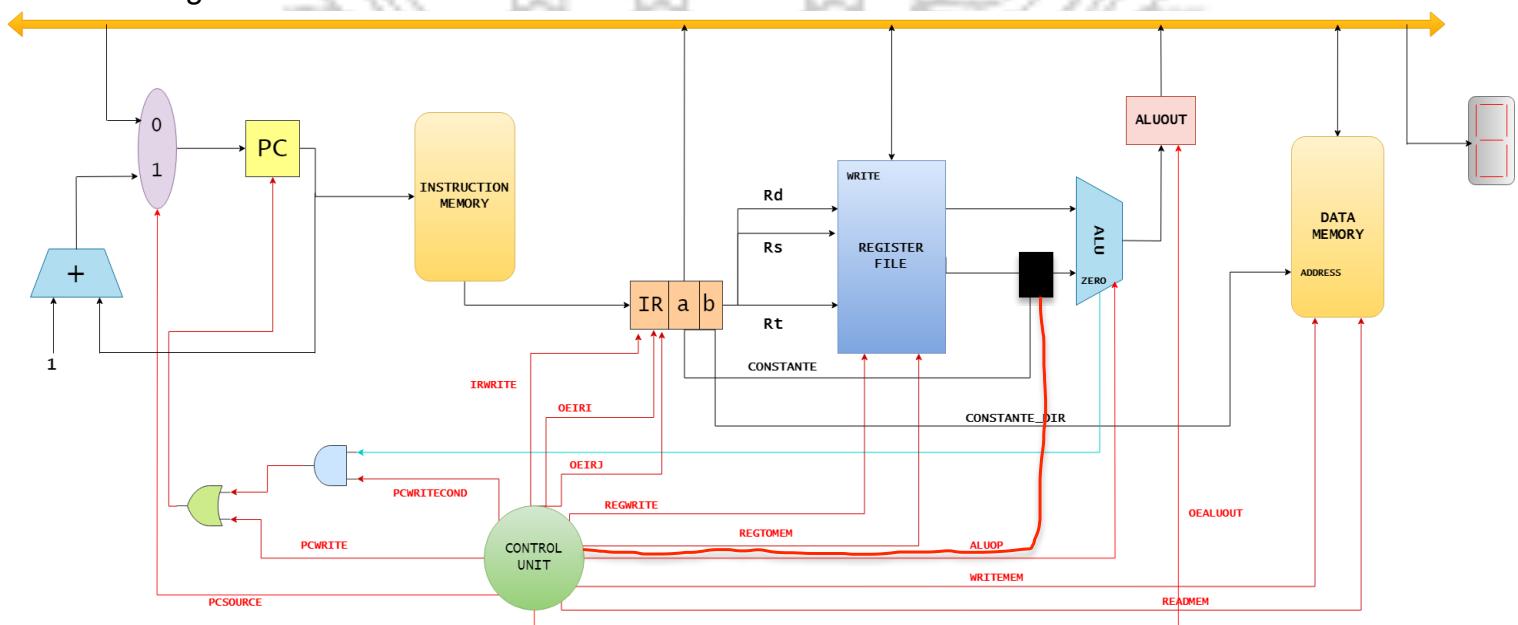
//CARGAR DATOS DE REGISTRO A MEMORIA
Input:
    beq sp,dirección,Input1
    addi sp,sp,1
    j Input
Input1:
    addi r1,r1,dato
    addi (sp),r1,#1
```

DIAGRAMA ESTRUCTURAL DE LA ARQUITECTURA DEL PROCESADOR

La arquitectura a implementar en el sistema de computo se basa en el funcionamiento de la arquitectura multiciclo de MIPS pero sin la necesidad de utilizar hardware que va a ser innecesario a la hora de ejecutar el programa. Por ende, se podría considerar un hibrido con otras arquitecturas antes vistas en el curso donde podremos encontrar nuevo hardware como un bus principal y un sumador de uno en uno encargado de aumentar el Program Counter (PC).



Para el manejo de la interfaz con el usuario y las diferentes salidas del programa se tendrá en cuenta el display 7 segmentos de la tarjeta FPGA en el cual se mostrarán los resultados de las diferentes funciones a realizar. Para ello, se logrará conectar el display al bus principal de la arquitectura donde estarán los datos luego de ser operados. A continuación, la imagen de la arquitectura junto con el display 7 segmentos.



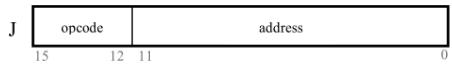
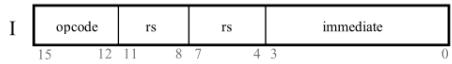
CAGD Reference Data



CORE INSTRUCTION SET

NAME, MNEMONIC	FOR- MAT	OPERATION	OPCODE
Add	add R	R[rd] = R[rs] + R[rt]	0000
Addi	addi I	R[rd] = R[rs] + 0	0001
Subtract	sub R	R[rd] = R[rs] - R[rt]	0010
Multiply	mul R	R[rd] = R[rs] * R[rt]	0011
Branch On Equal	beq I	If(R[rs]==R[rt]) PC=PC+1+BranchAddr	0100
Branch On Not Equal	bneq I	If(R[rs]!=R[rt]) PC=PC+1+BranchAddr	0101
Branch Less Than	blt I	If(R[rs]<R[rt]) PC=Label	0110
Branch Greater Than	bgt I	If(R[rs]>R[rt]) PC=Label	0111
Branch Greater Than Or Equal	bgte I	If(R[rs]>=R[rt]) PC=Label	1000
Branch Less Than Or Equal To Zero	bgtz I	If(R[rs]<R[rt]) PC=Label	1001
Jump	j J	PC = JumpAddr	1010
Input	in I		1011
Output	out I		1100

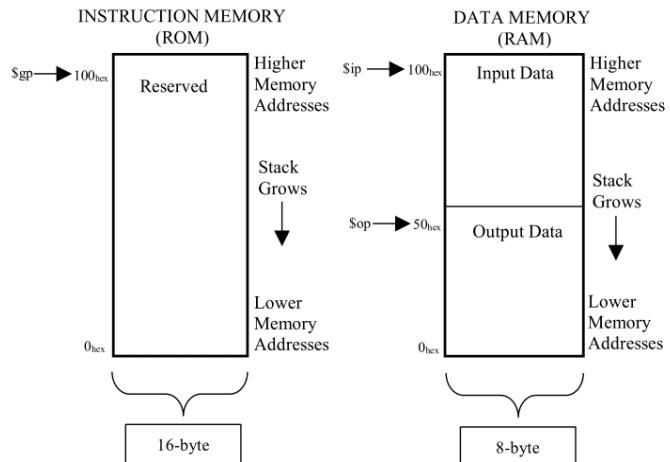
BASIC INSTRUCTION FORMATS



OPCODES, BASE CONVERSION, ASCII SYMBOLS

CAGD Opcode (15:12)	Binary	Decimal	Hexadecimal	ASCII Character
add	0000	0	0	NUL
addi	0001	1	1	SOH
sub	0010	2	2	STX
mul	0011	3	3	ETX
beq	0100	4	4	EOT
bneq	0101	5	5	ENQ
blt	0110	6	6	ACK
bgt	0111	7	7	BEL
bgte	1000	8	8	BS
bgtz	1001	9	9	HT
j	1010	10	a	LF
in	1011	11	b	VT
out	1100	12	c	FF

MEMORY ALLOCATION



REGISTER NAME, NUMBER, USE

NAME	NUMBER	USE
\$0	0	The Constant Value 0
\$at	1	Assembler Temporary
\$ip	2	Pointer for Input
\$op	3	Pointer for Output
\$r0-\$r7	4-11	Temporaries
\$gp	12	Global Pointer

BIBLIOGRAFIA

Lady-Comp (2018). El ciclo menstrual y la ovulación. Recuperado de
<http://www.lady-comp.es/ciclo-menstrual>

Glow. (Año 2011). Eve (Versión 2.12.1) [Aplicación Móvil].

MIPS Reference Data (Green Card). Elsevier, Inc.

