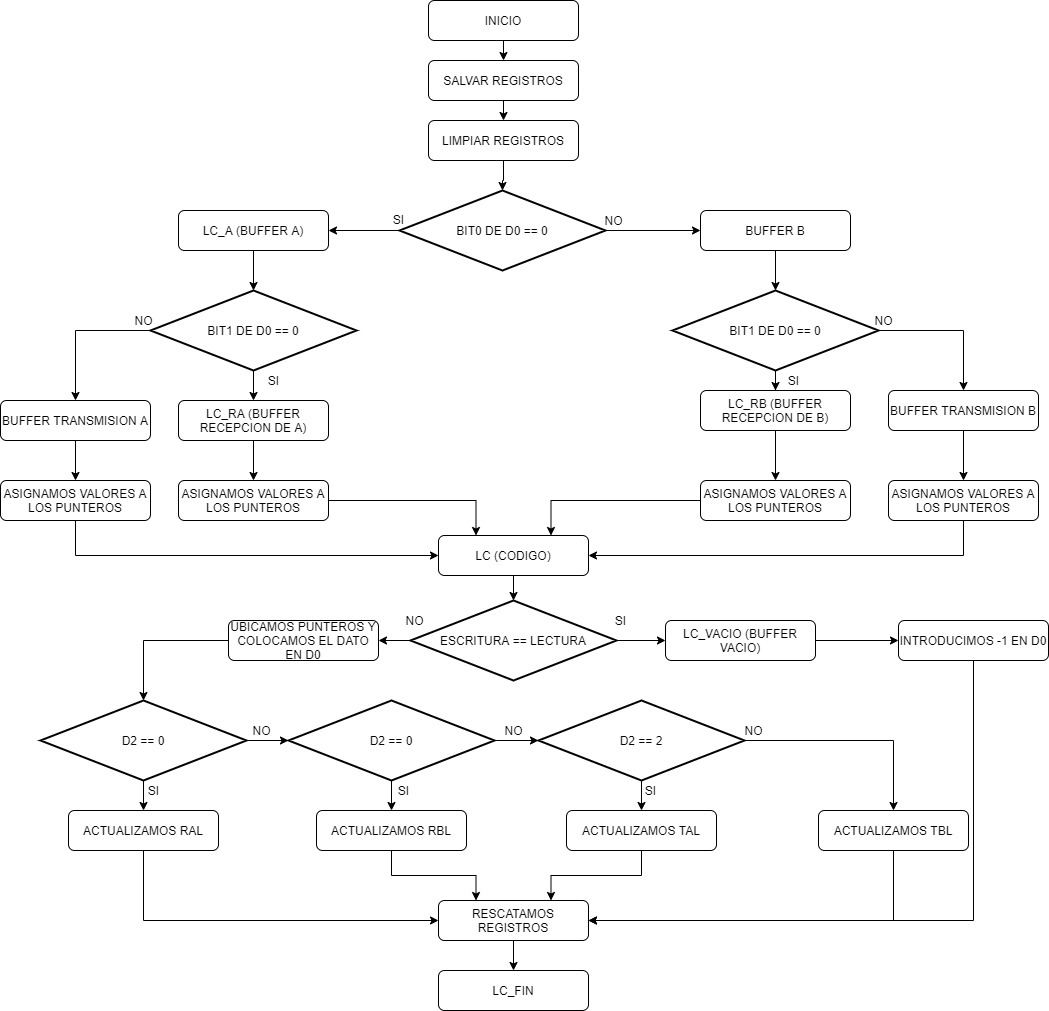
Universidad Politécnica de Madrid

**Proyecto entrada salida: memoria**

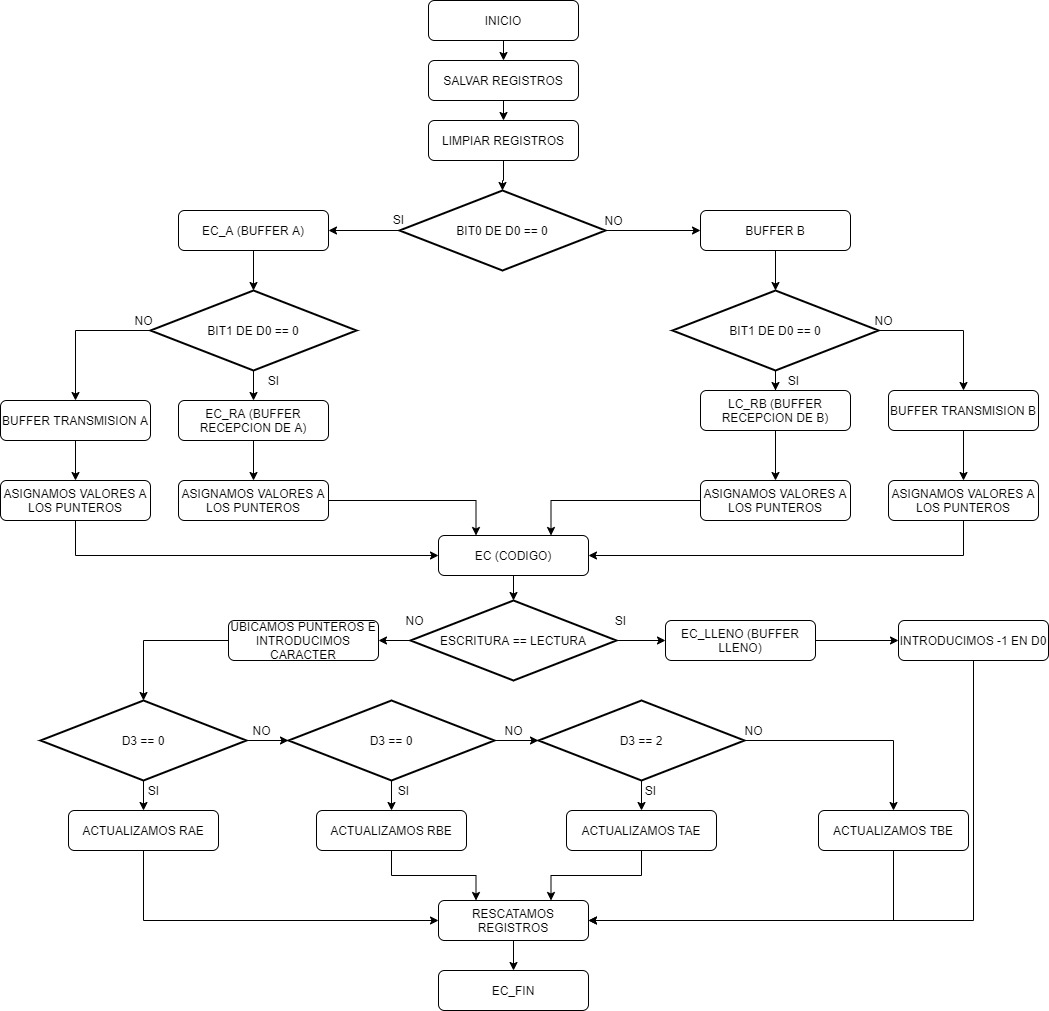
Autores: Jiménez Pérez Juan, N:190204

21 de mayo del 2021

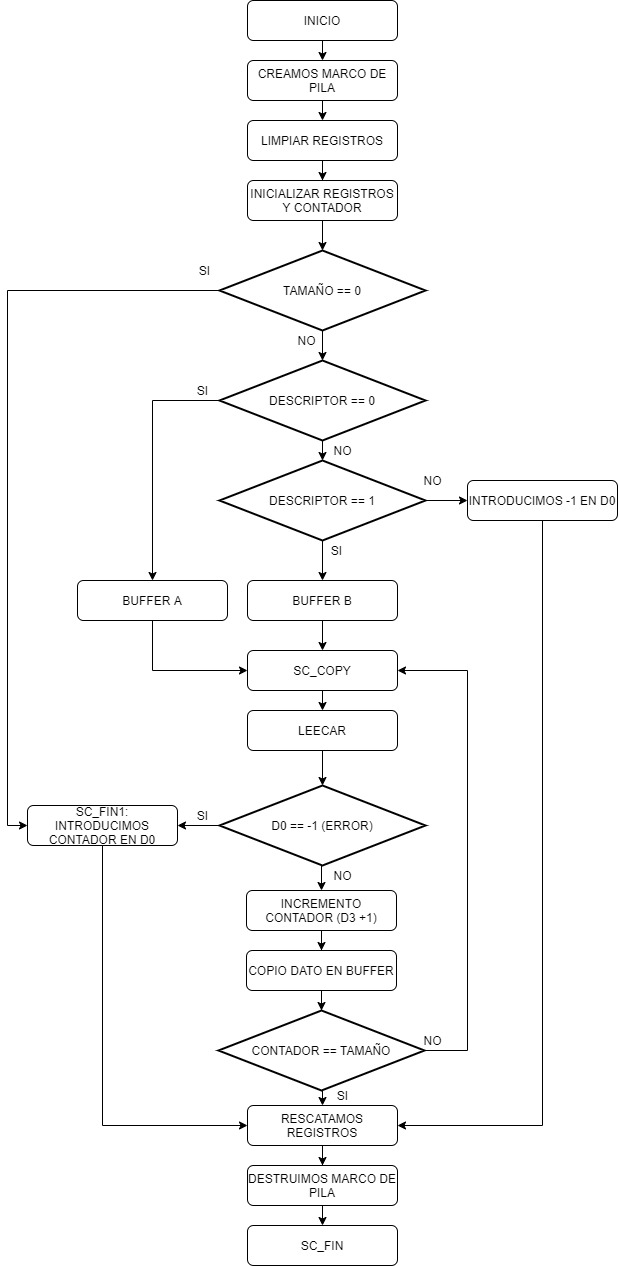
1. **Diagramas de flujo**
   1. **LEECAR**



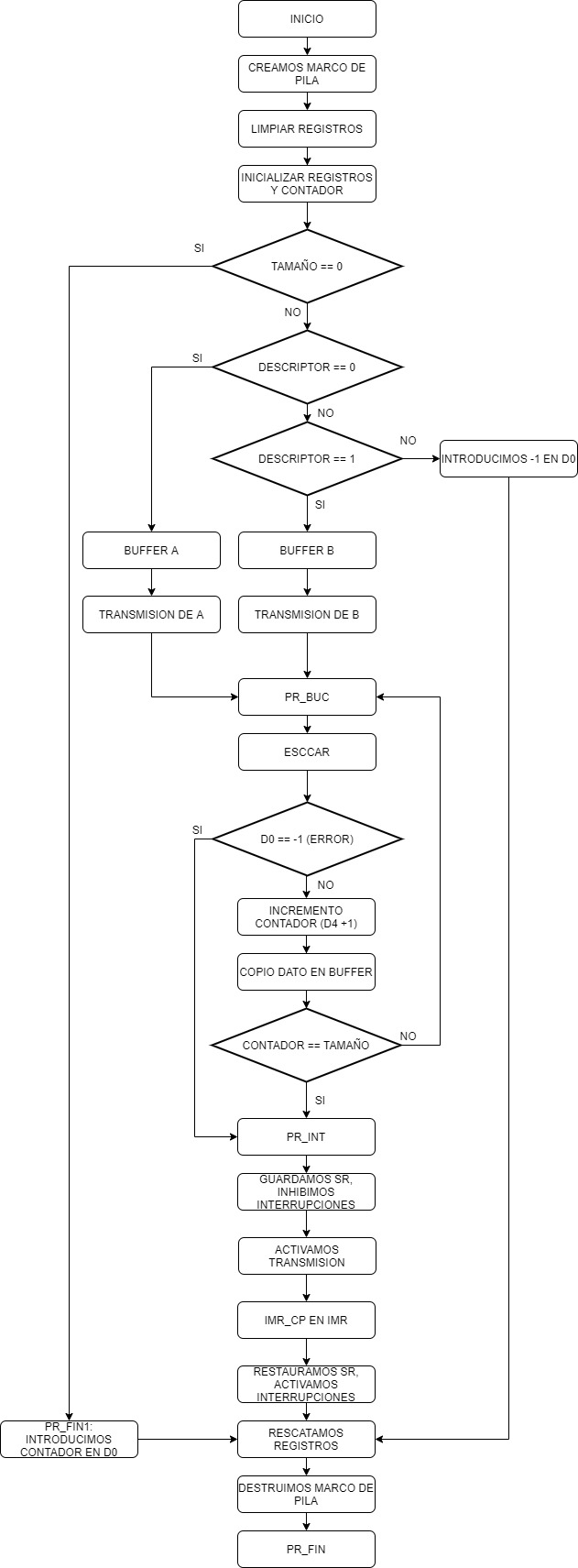
* 1. **ESCCAR**

****

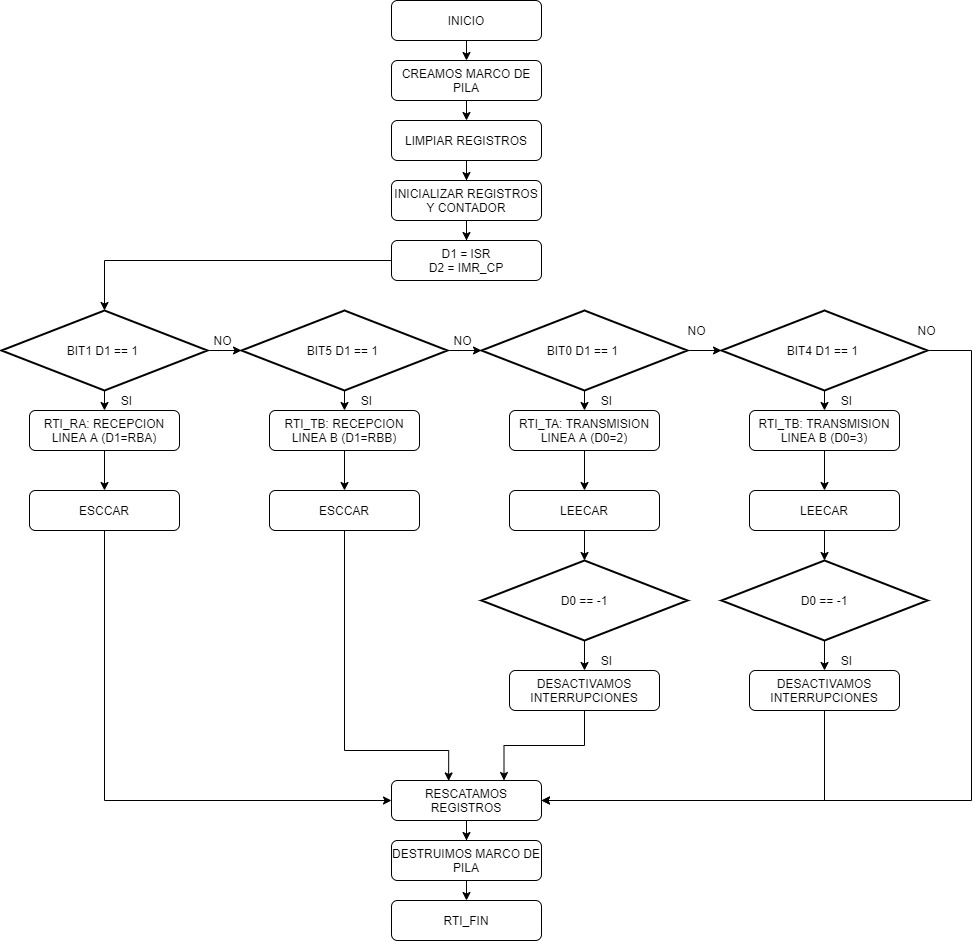
* 1. **SCAN**

****

* 1. **PRINT**

****

* 1. **RTI**

****

1. **Listado de comentarios:**
   1. **INIT:**

MOVE.B #%00010000,CRA \*Reinicia el puntero MR1

MOVE.B #%00000011,MR1A \*8 bits por caracter.

MOVE.B #%00000000,MR2A \*Eco desactivado.

MOVE.B #%11001100,CSRA \*Velocidad = 38400 bps.

MOVE.B #%00000000,ACR \*Velocidad = 38400 bps.

MOVE.B #%00000101,CRA \*transmisión y recepción activados.

MOVE.B #%00010000,CRB \*Reinicia el puntero MR1

MOVE.B #%00000011,MR1B \*8 bits por caracter.

MOVE.B #%00000000,MR2B \*Eco desactivado.

MOVE.B #%11001100,CSRB \*Velocidad = 38400 bps.

MOVE.B #%00000101,CRB \*transmisión y recepción activados.

MOVE.B #%01000000,IVR \*Vector de Interrupción 40

MOVE.B #%00100010,IMR \*IMR

MOVE.B #%00100010,IMR\_CP \*IMR\_CP

MOVE.L #RTI,$100 \*RTI

MOVEM.L A0/D0-D1,-(A7) \*Salvamos los registros

MOVE.L #TAMANO,D0

SUB.L #1,D0

MOVE.L #recep\_A,rae

MOVE.L #recep\_A,ral

MOVE.L #recep\_A,A0

MOVE.L A0,D1 \*En D1 esta 0x400

ADD.L D0,D1 \*Puntero al final

MOVE.L D1,raf

MOVE.L #recep\_B,rbe

MOVE.L #recep\_B,rbl

MOVE.L #recep\_B,A0

MOVE.L A0,D1 \*En D1 está 0xbd1

ADD.L D0,D1 \*puntero al final

MOVE.L D1,rbf

MOVE.L #trans\_A,tae

MOVE.L #trans\_A,tal

MOVE.L #trans\_A,A0

MOVE.L A0,D1 \*En d1 esta 0x13a2

ADD.L D0,D1 \*puntero al final

MOVE.L D1,taf

MOVE.L #trans\_B,tbe

MOVE.L #trans\_B,tbl

MOVE.L #trans\_B,A0

MOVE.L A0,D1 \*En d1 esta 0x1b73

ADD.L D0,D1 \*puntero al final

MOVE.L D1,tbf

MOVEM.L (A7)+,A0/D0-D1 \*Rescatamos registros

RTS

* 1. **LEECAR:**

MOVEM.L A1-A4/D1-D2,-(A7) \*Salvamos los registros

MOVE.L #0,A1 \*Limpiamos todos los registros a utilizar

MOVE.L #0,A2

MOVE.L #0,A3

MOVE.L #0,A4

MOVE.L #0,D1

MOVE.L #0,D2

MOVE.L #TAMANO,D1 \*TAMANO en D1

BTST #0,D0 \*Consulta bit 0

BEQ LC\_A \*De ser 0, el buffer es el A, si no, continua con el B

BTST #1,D0 \*Consulta el contenido del bit 1

BEQ LC\_RB \*De ser 0 es de recepción (recep\_B), si no,

continua con transmisión (trans\_B)

\*Trans\_B:

MOVE.L tbe,A1 \*Escritura

MOVE.L tbl,A2 \*Lectura

MOVE.L tbf,A3 \*Fin

MOVE.L #3,D2 \*3: indica que se desea acceder al buffer

interno de transmisión de la línea B.

BRA LC

\*Recep\_B

LC\_RB:

MOVE.L rbe,A1 \*Escritura

MOVE.L rbl,A2 \*Lectura

MOVE.L rbf,A3 \*Fin

MOVE.L #1,D2 \*1: indica que se desea acceder al buffer

interno de recepción de la línea B.

BRA LC

LC\_A:

BTST #1,D0 \*Consulta bit 1

BEQ LC\_RA \*De ser 0 es de recepción (recep\_A), si no,

continua con transmisión (trans\_A)

\*Trans\_A

MOVE.L tae,A1 \*Escritura

MOVE.L tal,A2 \*Lectura

MOVE.L taf,A3 \*Fin

MOVE.L #2,D2 \*2: indica que se desea acceder al buffer

interno de transmisión de la línea A.

BRA LC

\*Recep\_A

LC\_RA:

MOVE.L rae,A1 \*Escritura

MOVE.L ral,A2 \*Lectura

MOVE.L raf,A3 \*Fin

MOVE.L #0,D2 \*0: indica que se desea acceder al buffer

interno de recepción de la línea A.

LC:

CMP A1,A2 \*Compara escritura y lectura

BEQ LC\_VACIO \*Si son iguales, el buffer esta vacío

MOVE.L #0,D0 \*Inicializamos D0

MOVE.B (A2),D0 \*Coloca el dato en D0

MOVE.L A2,A4 \*Puntero auxiliar A4

ADD.L #1,A4 \*A4 + 1

CMP A3,A2 \*Compara si el puntero de lectura está en el

final del buffer

BNE LC\_0 \*Si no está, actualizamos punteros

SUB.L D1,A4 \*Si esta, lo ubicamos en el inicio

LC\_0:

CMP.L #0,D2 \*recepción de A

BNE LC\_1

MOVE.L A4,ral \*Actualizamos ral

BRA LC\_FIN

LC\_1:

CMP.L #1,D2 \*recepción de B

BNE LC\_2

MOVE.L A4,rbl \*Actualizamos rbl

BRA LC\_FIN

LC\_2:

CMP.L #2,D2 \*transmisión de A

BNE LC\_3

MOVE.L A4,tal \*Actualizamos tal

BRA LC\_FIN

LC\_3:

MOVE.L A4,tbl \*Actualizamos tbl

BRA LC\_FIN

LC\_VACIO:

MOVE.L #0,D0 \*Inicializamos D0

MOVE.L #$ffffffff,D0 \*Metemos un -1 en D0

LC\_FIN:

MOVEM.L (A7)+,A1-A4/D1-D2 \*Rescatamos registros

RTS

* 1. **ESCCAR**

MOVEM.L A1-A4/D1-D3,-(A7) \*Salvamos los registros

MOVE.L #0,A1 \*Limpiamos todos los registros a utilizar

MOVE.L #0,A2

MOVE.L #0,A3

MOVE.L #0,A4

MOVE.L #0,D2

MOVE.L #0,D3

MOVE.L #TAMANO,D2 \*TAMANO en D2

BTST #0,D0 \*Consulta bit 0

BEQ EC\_A \*De ser 0, el buffer es el A, si no, continua con

el B

BTST #1,D0 \*Consulta bit 1

BEQ EC\_RB \*De ser 0 es de recepción (recep\_B), si no,

continua con transmisión (trans\_B)

\*Transimision\_B

MOVE.L tbe,A1 \*Escritura

MOVE.L tbl,A2 \*Lectura

MOVE.L tbf,A3 \*Fin

MOVE.L #3,D3 \*3: indica que se desea acceder al buffer

interno de transmisión de la línea B.

BRA EC

\*Recep\_B

EC\_RB:

MOVE.L rbe,A1 \*Escritura

MOVE.L rbl,A2 \*Lectura

MOVE.L rbf,A3 \*Fin

MOVE.L #1,D3 \*1: indica que se desea acceder al buffer

interno de recepción de la línea B.

BRA EC

EC\_A:

BTST #1,D0 \*Consulta bit 1

BEQ EC\_RA \*De ser 0 es de recepción (recep\_A), si no,

continua con transmisión (trans\_A)

\*Trans\_A

MOVE.L tae,A1 \*Escritura

MOVE.L tal,A2 \*Lectura

MOVE.L taf,A3 \*Fin

MOVE.L #2,D3 \*2: indica que se desea acceder al buffer

interno de transmisión de la línea A.

BRA EC

\*Recep\_A

EC\_RA:

MOVE.L rae,A1 \*Escritura

MOVE.L ral,A2 \*Lectura

MOVE.L raf,A3 \*Fin

MOVE.L #0,D3 \*0: indica que se desea acceder al buffer

interno de recepción de la línea A.

EC:

MOVE.L A1,A4 \*Puntero auxiliar A4

ADD.L #1,A4 \*A4 + 1

CMP.L A1,A3 \*Comprobamos si estamos en el fin

BNE EC\_0

SUB.L D2,A4 \*Apunta al inicio

EC\_0:

CMP.L A4,A2 \*Escritura y Lectura iguales?

BEQ EC\_LLENO \*Si z=1 salto a EC\_LLENO

MOVE.B D1,(A1) \*Introduzco el carácter

MOVE.L #0,D0 \*Reinstauro D0 a 0

CMP.L #0,D3 \*recepción de A

BNE EC\_1

MOVE.L A4,rae \*Actualizamos rae

BRA EC\_FIN

EC\_1:

CMP.L #1,D3 \*recepción de B

BNE EC\_2

MOVE.L A4,rbe \*Actualizamos rbe

BRA EC\_FIN

EC\_2:

CMP.L #2,D3 \*Transmisión de A

BNE EC\_3

MOVE.L A4,tae \*Actualizamos tae

BRA EC\_FIN

EC\_3:

MOVE.L A4,tbe \*Actualizamos tbe

BRA EC\_FIN

EC\_LLENO:

MOVE.L #0,D0 \*Inicializamos D0

MOVE.L #$ffffffff,D0 \*Metemos un -1 en D0

EC\_FIN:

MOVEM.L (A7)+,A1-A4/D1-D3 \*Rescatamos registros

RTS

* 1. **SCAN**

LINK A6,#-16 \*Creamos el marco de pila para guardar los registros

MOVE.L A1,-4(A6) \*Salvamos los registros

MOVE.L D1,-8(A6)

MOVE.L D2,-12(A6)

MOVE.L D3,-16(A6)

MOVE.L #0,D0 \*Limpiamos todos los registros a utilizar

MOVE.L #0,D1

MOVE.L #0,D2

MOVE.L #0,D3 \*Inicializamos contador

MOVE.L #0,A1

MOVE.L 8(A6),A1 \*Buffer

MOVE.W 12(A6),D1 \*Descriptor

MOVE.W 14(A6),D2 \*Tamaño

CMP.W #0,D2 \*Comprobamos si el tamaño es 0, si es así

termina la subrutina

BEQ SC\_FIN1

CMP.W #0,D1 \*Comprobamos el descriptor

BEQ SC\_COPY \*Si el descriptor es 0 saltamos a SC\_A

CMP.W #1,D1

BEQ SC\_COPY \*Si descriptor es 1 saltamos a SC\_B.

MOVE.L #$ffffffff,D0 \*En caso de no cumplirse ninguna de las

anteriores devolvemos error

JMP SC\_FIN

SC\_COPY:

MOVE.W 12(A6),D0 \*Ponemos en D0 el descriptor

BSR LEECAR \*Lee el dato y lo devuelve en D0

CMP.L #$ffffffff,D0 \*Comprueba si D0 ha devuelto error, si ha

devuelto error entonces salto a SC\_FIN

BEQ SC\_FIN1

ADD.L #1,D3 \*Incrementamos el contador

MOVE.B D0,(A1)+ \*Copiamos el dato leído en el buffer y

aumentamos su dirección

CMP.L D3,D2 \*Comprobamos si el contador es igual al

tamaño

BNE SC\_COPY \*Si no son iguales, volvemos a SC\_COPY

SC\_FIN1:

MOVE.L D3,D0

SC\_FIN:

MOVE.L -4(A6),A1 \*Rescatamos registros

MOVE.L -8(A6),D1

MOVE.L -12(A6),D2

MOVE.L -16(A6),D3

UNLK A6 \*Destruimos marco de pila.

RTS \*Retorno.

* 1. **PRINT**

LINK A6,#-20 \*Creamos el marco de pila para guardar los

registros

MOVE.L A1,-4(A6) \*Salvamos los registros

MOVE.L D1,-8(A6)

MOVE.L D2,-12(A6)

MOVE.L D3,-16(A6)

MOVE.L D4,-20(A6)

MOVE.L #0,A1 \*Limpiamos todos los registros a utilizar

MOVE.L #0,D1

MOVE.L #0,D2

MOVE.L #0,D3

MOVE.L #0,D4 \*Inicializamos contador

MOVE.L 8(A6),A1 \*Buffer

MOVE.W 12(A6),D1 \*Descriptor

MOVE.W 14(A6),D2 \*Tamaño

CMP.W #0,D2 \*Si el tamaño es 0 se termina la subrutina

BEQ PR\_FIN1

CMP.W #0,D1 \*Si 0:

BEQ PR\_A \*Buffer de Transmisión de A

CMP.W #1,D1 \*Si 1:

BEQ PR\_B \*Buffer de Transmisión de B

MOVE.L #$ffffffff,D0 \*De no ser ninguno, se introduce -1 (Error) y se

sale de la subrutina

JMP PR\_FIN

PR\_A:

MOVE.L #2,D3 \*Se mete un 2 en D3 para acceder al buffer de

Transmisión de A

JMP PR\_BUC

PR\_B:

MOVE.L #3,D3 \*Se mete un 3 en D3 para acceder al buffer de

Transmisión de B

PR\_BUC:

MOVE.B (A1)+,D1 \*Extraemos el dato y avanzamos el puntero del

buffer

MOVE.L D3,D0 \*Copiamos D3 en D0 para pasarlo como

parámetro a ESCCAR

BSR ESCCAR

CMP #$ffffffff,D0 \*Comprobamos si hubo error en ESCCAR (si hay

nos salimos)

BEQ PR\_INT

ADD.L #1,D4 \*Incrementamos el contador

MOVE.W 14(A6),D2

CMP.L D4,D2 \*Contador y tamaño iguales termina la

subrutina

BNE PR\_BUC

PR\_INT:

CMP.L #0,D4

BEQ PR\_FIN1

MOVE.L #0,D1 \*Limpiamos D1

MOVE.W SR,D1 \*Guardamos SR en D1

MOVE.W #$2700,SR \*Inhibimos las interrupciones

CMP #2,D3 \*Si 2:

BEQ PR\_IMRA \*Buffer de Transmisión de A

BSET #4,IMR\_CP \*Como estamos en el buffer de transmisión B

activamos el bit 4

JMP PR\_IMR

PR\_IMRA:

BSET #0,IMR\_CP \*Como estamos en el buffer de transmisión A

activamos el bit 0

PR\_IMR:

MOVE.B IMR\_CP,IMR \*Copiamos IMR\_CP a IMR

MOVE.W #$2000,SR \*Activamos de nuevo las interrupciones

MOVE.W D1,SR \*Restituimos el SR al valor que tenía

previamente

PR\_FIN1:

MOVE.L D4,D0 \*Movemos el contador a D0

PR\_FIN:

MOVE.L -4(A6),A1 \*Rescatamos registros

MOVE.L -8(A6),D1

MOVE.L -12(A6),D2

MOVE.L -16(A6),D3

MOVE.L -20(A6),D4

UNLK A6 \*Destruimos el marco de pila

RTS

* 1. **RTI**

LINK A6,#-12 \*Creamos el marco de pila para guardar los

registros

MOVE.L D1,-4(A6) \*Salvamos los registros

MOVE.L D2,-8(A6)

MOVE.L D0,-12(A6)

MOVE.L #0,D0 \*Limpiamos todos los registros a utilizar

MOVE.L #0,D1

MOVE.L #0,D2

MOVE.B ISR,D1 \*ISR en D1

MOVE.B IMR\_CP,D2 \*IMR\_CP en D2

AND.B D2,D1 \*Vemos que periférico interrumpe

BTST #1,D1 \*Si bit 1 RxRDYA/FFULLA = 1:

BNE RTI\_RA \*recepción línea A

BTST #5,D1 \*Si bit 5 RxRDYB/FFULLB = 1:

BNE RTI\_RB \*recepción línea B

BTST #0,D1 \*Si bit 0 TxRDYA = 1:

BNE RTI\_TA \*transmisión línea A

BTST #4,D1 \*Si bit 4 TxRDYB = 1:

BNE RTI\_TB \*transmisión línea B

JMP RTI\_FIN

RTI\_TA:

MOVE.L #2,D0 \*Descriptor a 2

BSR LEECAR \*Llamamos a LEECAR

CMP.L #$ffffffff,D0 \*Si D0 es -1:

BEQ RTI\_TAI \*Desactivamos las interrupciones de la línea de

transmisión A

MOVE.B D0,TBA \*El carácter leído lo movemos a TBA

JMP RTI\_FIN

RTI\_TAI:

BCLR #0,IMR\_CP \*Desactivamos el bit 0 de IMR\_CP

MOVE.B IMR\_CP,IMR \*Copiamos IMR\_CP a IMR

JMP RTI\_FIN

RTI\_RA:

MOVE.L #0,D1 \*Limpiamos D1

MOVE.B RBA,D1 \*RBA en D1

MOVE.L #0,D0 \*Buffer A de recepción en ESCCAR

BSR ESCCAR \*Llamamos a ESCCAR

JMP RTI\_FIN

RTI\_TB:

MOVE.L #3,D0 \*Descriptor a 3

BSR LEECAR \*Llamamos a LEECAR

CMP.L #$ffffffff,D0 \*Si D0 es -1:

BEQ RTI\_TBI \*Desactivamos las interrupciones de la línea de

transmisión B

MOVE.B D0,TBB \*El carácter leído lo movemos a TBB

JMP RTI\_FIN

RTI\_TBI:

BCLR #4,IMR\_CP \*Desactivamos el bit 4 de IMR\_CP

MOVE.B IMR\_CP,IMR \*Copiamos IMR\_CP a IMR

JMP RTI\_FIN

RTI\_RB:

MOVE.L #0,D1 \*Limpiamos D1

MOVE.B RBB,D1 \*RBB en D1

MOVE.L #1,D0 \*Buffer B de recepción en ESCCAR

BSR ESCCAR \*Llamamos a ESCCAR

JMP RTI\_FIN

RTI\_FIN:

MOVE.L -4(A6),D1 \*Rescatamos registros

MOVE.L -8(A6),D2

MOVE.L -12(A6),D0

UNLK A6 \*Destruimos el marco de pila

RTE

1. **Descripción del juego de ensayo**

El juego de ensayo fue diseñado basándonos en los casos de prueba que se encontraban en el manual. Nos guiamos por él e íbamos haciendo los cambios que considerábamos oportunos para poder probar casos específicos de ciertas subrutinas. Los cambios generalmente eran con respecto al tamaño de los datos que introducíamos y lógicamente cual de las dos líneas se utilizaban. Con las pruebas iniciales, en las que los tamaños eran pequeños usábamos tanto la línea A como la B para introducir datos, pero ya en las pruebas finales en las que los tamaños de los bloques que querían leerse eran más grandes, usábamos traza.log para ver los movimientos que había en memoria y poder depurar así el código.

1. **Observaciones finales y comentarios personales**

En general fue un practica con un nivel de dificultad bastante alto. Requería un conocimiento completo sobre el funcionamiento de las interrupciones, pero luego de entenderlo, era posible completarla sin problema.

Una de las subrutinas que mas nos dificulto la entrega fue la de print, esto fue ya que teníamos confusión con el funcionamiento de IMR y con el hecho de activar/desactivar las interrupciones. Leyendo detalladamente el manual y probando logramos sacarlo finalmente. Utilizar el caso de prueba y las funciones del bsvc pudimos probar varios casos a lo largo del proyecto y esto facilito el proceso. Al comienzo costó un poco acostumbrarnos a como funcionaba todo y entender que hacía que, pero poco a poco fuimos entendiendo (conjunto al manual y los videos de explicación).

Esccar nos pareció la más fácil debido a que es bastante parecida a Leecar, con algún que otro cambio. Al tener ya lista leecar y entender cómo funcionaba fue mucho más simple hacerla. En cambio, Print fue la más compleja puesto que una vez que habías escrito todos los caracteres en su línea correspondiente, tenias que activar el bit que correspondiera de IMR y como hemos mencionado antes, no comprendíamos muy bien cómo funcionaba, así que nos costo un tiempo ver como arreglarlo.

El trabajo lo realizamos utilizando unas aplicaciones muy útiles como “Discord” y “code together”. Estas nos permitieron realizar el trabajo por completo a distancia, sobre todo la segunda que nos dejaba editar el código a la vez mientras utilizábamos la primera para hablar y en algunos casos compartir nuestras pantallas. También cabe destacar que de esta misma manera hicimos el proyecto de Estructura de Computadores así que ya estábamos acostumbrados a trabajar de esta manera y teníamos buena química en ello.