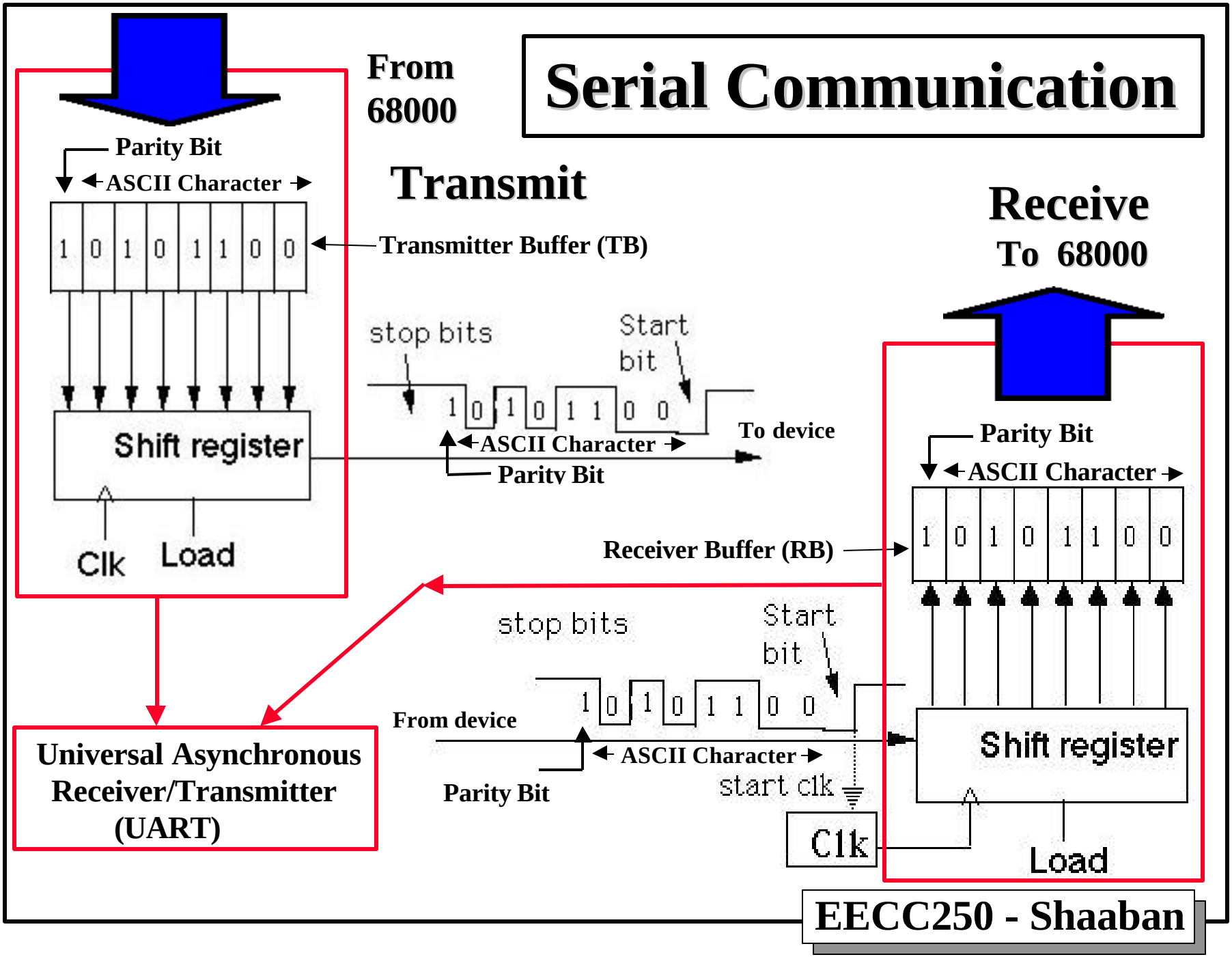
Serial Communication



ASCII Code Table

HEX	DEC	CHR	Ctrl
00	0	NUL	^@
01	1	SOH	^A
02	2	STX	^B
03	3	ETX	^C
04	4	EOT	^D
05	5	ENQ	^E
06	6	ACK	^F
07	7	BEL	^G
08	8	BS	^H
09	9	HT	^I
0A	10	LF	^J
0B	11	VT	^K
0C	12	FF	^L
0D	13	CR	^M
0E	14	SO	^N
0F	15	SI	^O
10	16	DLE	^P
11	17	DC1	^Q
12	18	DC2	^R
13	19	DC3	^S
14	20	DC4	^T
15	21	NAK	^U
16	22	SYN	^V
17	23	ETB	^W
18	24	CAN	^X
19	25	EM	^Y
1A	26	SUB	^Z
1B	27	ESC	
1C	28	FS	
1D	29	GS	
1E	30	RS	
1F	31	US	

HEX	DEC	CHR
20	32	SP
21	33	!
22	34	”
23	35	#
24	36	$
25	37	%
26	38	&
27	39	’
28	40	(
29	41	)
2A	42	*
2B	43	+
2C	44	,
2D	45	-
2E	46	.
2F	47	/
30	48	0
31	49	1
32	50	2
33	51	3
34	52	4
35	53	5
36	54	6
37	55	7
38	56	8
39	57	9
3A	58	:
3B	59	;
3C	60	<
3D	61	=
3E	62	>
3F	63	?

HEX	DEC	CHR
40	64	@
41	65	A
42	66	B
43	67	C
44	68	D
45	69	E
46	70	F
47	71	G
48	72	H
49	73	I
4A	74	J
4B	75	K
4C	76	L
4D	77	M
4E	78	N
4F	79	O
50	80	P
51	81	Q
52	82	R
53	83	S
54	84	T
55	85	U
56	86	V
57	87	W
58	88	X
59	89	Y
5A	90	Z
5B	91	[
5C	92	\
5D	93	]
5E	94	^
5F	95	_

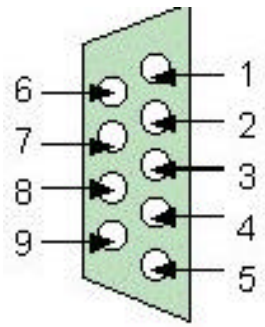
HEX	DEC	CHR
60	96	`
61	97	a
62	98	b
63	99	c
64	100	d
65	101	e
66	102	f
67	103	g
68	104	h
69	105	I
6A	106	j
6B	107	k
6C	108	l
6D	109	m
6E	100	n
6F	111	o
70	112	p
71	113	q
72	114	r
73	115	s
74	116	t
75	117	u
76	118	v
77	119	w
78	120	x
79	121	y
7A	122	z
7B	123	{
7C	124	|
7D	125	}
7E	126	~
7F	127	DEL

Full-Duplex Serial Communication Lines

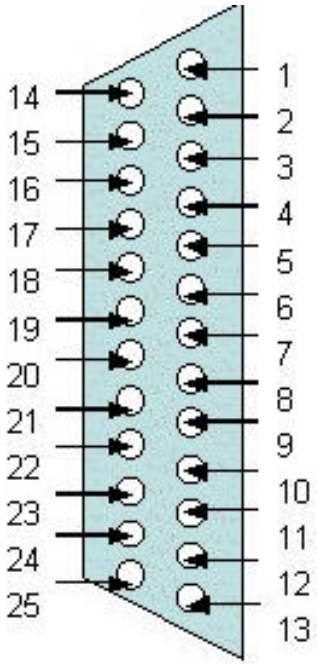
- **External Receiver Lines:**
  - **RxRTS* Low if local device can receive a character (connected to remote CTS*)**
  - **RxD Actual serial data received from remote device on this line.**
- **Transmitter Lines:**
  - **TxRTS Request To Send: indicates local device is ready to transmit a character**
  - **TxD Actual serial data transmitted to remote device on this line**
  - **CTS* Cleared To Send: Low indicates remote device ready to receive a character (connected to remote RxRTS)**

RS232C Serial Data Interface

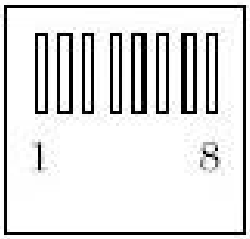
- **RS232C is the most commonly used serial data interface in the computer industry.**
- **The following diagrams and table give the pinout details for the four most commonly used physical serial data connectors:**



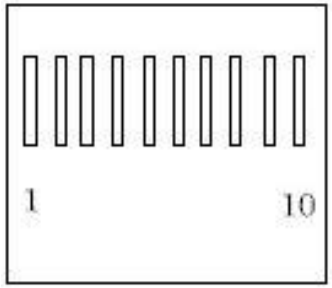
DB9



DB25



RJ45



RJ46

RS232C Serial Connectors Pinout

DB9	DB25	RJ45	RJ46	Signal	Usual Source
1	8	1	2	CD - Carrier Detect	MODEM
2	3	2	3	RxD - Receive Data	MODEM
3	2	3	4	TxD - Transmit Data	TERMINAL
4	20	4	5	DTR - Data Term'l Ready	TERMINAL
5	7	5	6	Signal Ground	
6	6	6	7	DSR - Data Set Ready	MODEM
7	4	7	8	RTS - Ready to Send	TERMINAL
8	5	8	9	CTS - Clear to Send	MODEM
9	22	9	10	RI - Ring Indication	MODEM
	1		1	Earth/Frame Ground	

Connector Pin #

Motorola 68681 Dual UART (DUART)

The Motorola 68681 Dual Universal Asynchronous Receiver/Transmitter (DUART) has the following features:

- **Two, independent, full-duplex asynchronous serial Receiver/Transmitter ports: A, B**
- **16, 8-bit registers for data buffering , control and status information.**
- **Each Receiver has a 4-byte buffer to hold incoming data.**
- **Independently programmable baud rate (bits per second) for each Receiver and Transmitter:**
  - **18 Fixed rates: 50 to 38400 baud**
- **Programmable data format allowing five to eight data bits.**
- **Programmable channel modes:**
  - **Normal (full-duplex).**
  - **Automatic echo.**
- **Versatile interrupt system:**
  - **Single interrupt output with four maskable interrupting conditions.**
  - **Interrupt vector output on interrupt acknowledge (IACK).**

68681 DUART Registers

Decimal Offset from DUART Base Address	Read	Write
0	Mode Register Port A (MR1A, MR2A)	Mode Register Port A (MR1A, MR2A)
2	Status Register Port A (SRA)	Clock Select Register Port A (CSRA)
4	Do not access	Command Register Port A (CRA)
6	Receiver Buffer Port A (RBA)	Transmitter Buffer Port A (TBA)
8	Input Port Change Register (IPCR)	Auxiliary Control Register (ACR)
10	Interrupt Status Register (ISR)	Interrupt Mask Register (IMR)
12	Current MSB of Counter (CUR)	Counter/Timer Upper Byte (CTUR)
14	Current LSB of Counter (CUL)	Counter/Timer Lower Byte (CTUL)
16	Mode Register Port B (MR1B,MR2B)	Mode Register Port B (MR1B,MR2B)
18	Status Register Port B (SRB)	Clock Select Register Port B (CSRB)
20	Do not access	Command Register Port B (CRB)
22	Receiver Buffer Port B (RBB)	Transmitter Buffer Port B (TBB)
24	Interrupt Vector Register (IVR)	Interrupt Vector Register (IVR)
26	Input Port (Unlatched)	Output Port Configuration (OPCR)
28	Start Counter Command	Output Port Bit Set
30	Stop Counter Command	Output Port Bit Clear

Port A Mode Register 1 (MR1A) and Port B Mode Register 1 (MR1B)

Rx RTS CONTROL	Rx IRQ Select	ERROR MODE	PARITY MODE	PARITY TYPE	Bits-per-character
Bit 7 0 = DISABLE 1 = ENABLE	Bit 6 0 = RxRDY 1 = FFULL	Bit 5 0 = CHAR 1 = BLOCK	Bit 4 Bit 3 0 0 = WITH PARITY 1 0 = NO PARITY	Bit 2 WITH PARITY 0= EVEN 1 = ODD	Bit 1 Bit 0 0 0 = 5 0 1 = 6 1 0 = 7 1 1 = 8

Port A Mode Register 2 (MR2A) and Port B Mode Register 2 (MR2B)

Port Mode	Tx RTS CONTROL	CTS ENABE TRANSMITTER	Stop Bit Length
Bit 7 Bit 6 0 0 = Normal 0 1 = Auto Echo 1 0 = LOCAL LOOPBACK 1 1 = REMOTE LOOPBACK	Bit 5 0 = DISABLE 1 = ENABLE	Bit 4 0 = DISABLE 1 = ENABLE	Bit 3 Bit 2 Bit 1 Bit 0 0 1 1 1 = 1 bit 1 1 1 1 = 2 bits

Clock-Select Register A (CSRA) and Clock-Select Register B (CSRB)

Receiver-Clock Select			Transmitter-Clock Select		
Bit 7 Bit 6 Bit 5 Bit 4	Set 1 ACR Bit 7 = 0	Set 2 ACR Bit 7 = 1	Bit 3 Bit 2 Bit 1 Bit 0	Set 1 ACR Bit 7 = 0	Set 2 ACR Bit 7 = 1
0 0 0 0	50	75	0 0 0 0	50	75
0 0 0 1	110	110	0 0 0 1	110	110
0 0 1 0	134.5	134.5	0 0 1 0	134.5	134.5
0 0 1 1	200	150	0 0 1 1	200	150
0 1 0 0	300	300	0 1 0 0	300	300
0 1 0 1	600	600	0 1 0 1	600	600
0 1 1 0	1200	1200	0 1 1 0	1200	1200
0 1 1 1	1050	2000	0 1 1 1	1050	2000
1 0 0 0	2400	2400	1 0 0 0	2400	2400
1 0 0 1	4800	4800	1 0 0 1	4800	4800
1 0 1 0	7200	1800	1 0 1 0	7200	1800
1 0 1 1	9600	9600	1 0 1 1	9600	9600
1 1 0 0	38400	19200	1 1 0 0	38400	19200
1 1 0 1	N/A	N/A	1 1 0 1	N/A	N/A
1 1 1 0	N/A	N/A	1 1 1 0	N/A	N/A
1 1 1 1	N/A	N/A	1 1 1 1	N/A	N/A

DUART+4 DUART+20

Port A Command Register (CRA) and Port B Command Register (CRB)

N/A	Misc Commands	Transmitter	Receiver
Bit 7	Bit 6 Bit 5 Bit 4	Bit 3 Bit 2	Bit 1 Bit 0
	0 0 0 = No Command 0 0 1 = Reset MR Pointer 0 1 0 = Reset Receiver 0 1 1 = Reset Transmitter 1 0 0 = Reset Error Status 1 0 1 = N/A 1 1 0 = N/A 1 1 1 = N/A	0 0 = No Action 0 1 = Transmitter Enabled 1 0 = Transmitter Disabled 1 1 = N/A	0 0 = No Action 0 1 = Receiver Enabled 1 0 = Receiver Disabled 1 1 = N/A

DUART+2 DUART+18

Port A Status Register (SRA) and Port B Status Register (SRB)

RECEIVED BREAK	FRAMING ERROR	PARITY ERROR	OVERRUN ERROR	TxEMT	TxRDY	FFULL	RxRDY
Bit 7 0 = No 1 = Yes	Bit 6 0 = No 1 = Yes	Bit 5 0 = No 1 = Yes	Bit 4 0 = No 1 = Yes	Bit 3 0 = No 1 = Yes	Bit 2 0 = No 1 = Yes	Bit 1 0 = No 1 = Yes	Bit 0 0 = No 1 = Yes

Auxiliary Control Register (ACR) DUART+8

Baud Rate Set	COUNTER/TIMER MODE AND SOURCE	DELTA IP3 IRQ	DELTA IP2 IRQ	DELTA IP1 IRQ	DELTA IP0 IRQ
Bit 7 0 = Set One 1 = Set Two	Bit 6 Bit 5 Bit 4 000 COUNTER MODE CLK ON IP2 011 COUNTER MODE INTERNAL CLK 100 TIMER MODE EXT CLK ON IP2 110 TIMER MODE INTERNAL CLK	Bit 3	Bit 2	Bit 1	Bit 0
			0 = DISABLE 1 = ENABLE		

Interrupt Vector Register (IVR)

DUART+24

Interrupt vector number: Bit7 - Bit0

Interrupt Status Register (ISR)

DUART+10 (read)

INPUT PORT CHANGE	DELTA BREAK B	RxRDYB FFULLB	COUNTER/ TIMER	TxRDYB	DELTA BREAK A	RxRDYA FFULLA	TxRDYA
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes	0 = No 1 = Yes

Interrupt Mask Register (IMR)

DUART+10 (write)

INPUT PORT CHANGE	DELTA BREAK B	RxRDYB FFULLB	COUNTER/ TIMER	TxRDYB	DELTA BREAK A	RxRDYA FFULLA	TxRDYA
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0 = Masked 1 = Pass	0 = Masked 1 = Pass	0 = Masked 1 = Pass	0 = Masked 1 = Pass	0 = Masked 1 = Pass	0 = Masked 1 = Pass	0 = Masked 1 = Pass	0 = Masked 1 = Pass

68681 DUART Registers Address Equates

DUART	EQU	$0FF800	Base Address of DUART
MRA	EQU	DUART+0	Mode Register Port A
SRA	EQU	DUART+2	Status Register Port A (read only).
CSRA	EQU	DUART+2	Clock Select Register Port A (write only)
CRA	EQU	DUART+4	Commands Register Port A (write only)
RBA	EQU	DUART+6	Receiver Buffer Port A (read only)
TBA	EQU	DUART+6	Transmitter Buffer Port A (write only)
ACR	EQU	DUART+8	Auxiliary Control Register
ISR	EQU	DUART+10	Interrupt Status Register (read only)
IMR	EQU	DUART+10	Interrupt Mask Register (write only)
MRB	EQU	DUART+16	Mode Register Port B
SRB	EQU	DUART+18	Status Register Port B (read only).
CSRB	EQU	DUART+18	Clock Select Register Port B (write only)
CRB	EQU	DUART+20	Commands Register Port B (write only)
RBB	EQU	DUART+22	Receiver Buffer Port B (read only)
TBB	EQU	DUART+22	Transmitter Buffer Port B (write only)
IVR	EQU	DUART+24	Interrupt Vector Register

Polled I/O Example Using Port A of DUART

- **Subroutine INITIAL, initializes port A of DUART to send and receive.**
- **Subroutine GET_CHAR inputs one character from port A to register D2 when port A receiver is ready with a character using busy-waiting**
- **Subroutine PUT_CHAR outputs one character to Port A transmitter from register D0 when port A transmitter is ready.**

```
*   DUART reset if needed
INITIAL     MOVE.B      #$30,CRA        Reset Port A transmitter
            MOVE.B      #$20,CRA        Reset Port A receiver
            MOVE.B      #$10,CRA        Reset Port A MR (mode register) pointer
*   Select baud rate, data format, and operating modes in ACR, MR1A, MR2A
            MOVE.B      #$80,ACR        Select baud rate set 2
            MOVE.B      #$BB,CSRA       Set both Rx, Tx speeds to 9600 baud
            MOVE.B      #$93,MRA        Set port A to 8 bit character, no parity
*                                       Enable RxRTS output using MR1A
            MOVE.B      #$37,MRA        Select normal operating mode
*                                       TxRTS, TxCTS, one stop bit using MR2A
            MOVE.B      #$05,CRA        Enable Port A transmitter and receiver
            RTS
```

Subroutines: GET_CHAR, PUT_CHAR

```
*  Subroutine GET_CHAR inputs a single character from  port  A receiver into
*   register D2  when port A receiver is ready using polling.
GET_CHAR   NOP
IN_POLL    MOVE.B   SRA,D1      Read port A status register SRA
           BTST     #0,D1       Test receiver ready bit RxRDY
           BEQ      IN_POLL     Wait until character is received
           MOVE.B   RBA,D2      Read character received  into D2
           RTS

*  Subroutine PUT_CHAR outputs a single character from  register D2  to port A
*   transmitter when Port A transmitter is  ready using polling.
PUT_CHAR   NOP
OUT_POLL   MOVE.B   SRA,D1      Read port A status register SRA
           BTST     #2,D1       Test transmitter ready bit TxRDY
           BEQ      OUT_POLL    Wait until transmitter A is ready
           MOVE.B   D0,TBA      Transmit character to port A
           RTS
```

Interrupt-Driven 68681 Terminal Character Input & Echo I/O Example

- A data terminal is connected to port A of the 68681 (receiver & transmitter).
- An interrupt should be generated every time a character is entered using the terminal keyboard.
- An interrupt service routine (ISR) for this interrupt should:
  - Store the character obtained from the terminal (using port A receiver) in a character buffer in memory at the address pointed to by A1
  - Echo the character to the terminal's screen (using transmitter of port A).
- The two routines needed: initialization subroutine A_INIT, and ISR, A_ISR

A_INIT → Point A1 to initial memory character buffer

→ Initializes Port A of 68681 to send and receive with interrupts enabled when a character is received.

→ Initialize Interrupt Vector Register and exception table entry

A_ISR → Store character in memory buffer

→ Wait until transmitter of port A is ready, then echo character to screen.

Interrupt-Driven 68681 I/O: A_INIT

```
A_VEC       EQU     64              Vector number for DUART interrupt
VEC_ADD     EQU     A_VEC*4         Interrupt vector table address
IMRM        EQU     %00000010
*  Initialization routine,  DUART assumed to have been reset else where
            ORG     $1000
A_INIT      LEA     BUFFER,A1       Initialize character buffer pointer
            LEA     A_ISR,A0
            MOVE.L  A0, VEC_ADD     Initialize exception  vector table entry
* Initialize port A of  DUART
            MOVE.B  #$13,MRA        Initialize MR1A
            MOVE.B  #$07,MRA        Initialize MR2A
            MOVE.B  #$BB,CSRA       Initialize CSRA
            MOVE.B  #$05,CRA        Initialize CRA
            MOVE.B  #$70,ACR        Initialize ACR
            MOVE.B  #A_VEC,IVR      Load interrupt vector in IVR
            MOVE.B  #IMRM, IMR      Initialize interrupt mask IMR
            RTS
            ORG $1500
BUFFER      DS.B    256
```

Interrupt-Driven 68681 I/O: A_ISR

```
*  Interrupt service routine A_ISR:
*      Make sure a character is actually available  in receive buffer A,  RBA,  otherwise return
*               If a received character  is found:
                        read RBA  and  store  in memory character buffer.
*                            then wait until  transmitter of  port A is  ready to transmit then:
*                                                 echo the character just received  to TBA
                ORG             $1200
A_ISR           MOVE.B          ISR,D0
                BTST.B          #1,D0           Verify a character is  available in RBA
                BEQ             DONE            If none return from interrupt
                MOVE.B          RBA,D1          Get character  in D1
                MOVE.B          D1,(A1)+        Put character in memory buffer
ECHO_W          MOVE.B          SRA,D2          Read port A status register SRA
                BTST            #2,D2           Test transmitter ready bit TxRDY
                BEQ             ECHO            If  not ready  wait
                MOVE.B          D1,TBA          Echo character on screen
DONE            RTE
```