CONFIDENTIAL

PDP-X Technical Memorandum #9

Title: Bus to Bus Adapter

Author(s): H. Burkhardt
L. Seligman

Index Keys: Bus
Communications
Interprocessor
IO
Peripherals

Distribution Key: A, B, C

Obsolete: None

Revision: None

Date: July 14, 1967

0. Overall Description

The Bus to Bus Adapter is a device used to connect two processors together through the IO Bus. It will be used for interprocessor communications in multiprocessor configurations. It will also be used for control communications between the arithmetic processor and IO processors. (See the Appendix ; PDP-X Technical Memorandum #6, pages 8-9, 20).

1. General Specifications

1.5 General Performance

The Bus to Bus Adapter consists of two receiver-transmitter pairs. This system is full duplex in that each pair is operated independently. Each pair is identical so that in the following only one of the pairs will be described. It is to be understood that the processor doing the receiving may simultaneously be transmitting.

Both the receiver and the transmitter have status registers that may be sensed to determine the operational state of the pair. In normal operations, the transmitter will transmit a command list to the receiver informing it of the quantity and nature of the following data. The receiver will process the command list, allocate a portion of its memory to receive this data and the transfer will begin. If, for some reason, the receiver cannot handle the quantity of data about to be sent by the transmitter, it may selectively receive portions of it or receive none of it. In either case, the data is not lost but is merely not sent until the receiver allows it to be.

The transmitter may be operated in either programmed mode or under control of the multiplexor channel. The same holds for the receiver.

The receiver-transmitter pair may be located together or may be separated if the distance between the two processors is significant. In this case, a separate transfer bus is placed between the transmitter and the receiver. See Appendix.

3.0 Programming

3.1 Instructions

3.1.1 Transmitter

The transmitter responds to IOC and IOX instructions to set or alter the SR (Status Register). The transmitter SR may be read or tested with IOS and IOT instructions. The IOW instruction may be used to transfer one byte of information to the transmitter data buffer. The BUSY bit in the SR is set, the REQ bit is cleared. If the REQ bit in the receiver is cleared, the transmitter data buffer is transferred to the receiver data buffer. Bits 9, 10 of the transmitter SR are transferred to bits 8, 9 of the receiver SR. The REQ bit in the receiver SR is set; the REQ bit in the transmitter SR is set and the BUSY bit in the transmitter SR is cleared. None of these events take place until the REQ bit in the receiver SR becomes a zero. If data is loaded into the transmitter data buffer during a multiplexor channel operation, the same events are repeated. When a channel overflow occurs, the LOW and UNUSUAL bits in the transmitter SR are set. When the data has been accepted by the receiver data buffer, REQ will be set in the transmitter SR and a low interrupt requested. The transmitter service routine can now respond to and service this interrupt. The Mode bit in the transmitter SR serves no function but to inform the receiver of the nature of the data that it has received.

3.1.2 Receiver

The receiver responds to IOC and IOX instructions to set or alter its SR (Status Register). The receiver SR may be read or tested with IOS and IOT instructions. Bits 8, 9, 11, 13 of the receiver SR may only be read or sensed. Attempts to alter these bits will have no effect. When the REQ bit in the receiver SR becomes a one, and IOR instruction may be used to read one byte of data. The SR contains the status of the data byte just read. The REQ bit should then be cleared to allow another data transfer from the transmitter data buffer to the receiver data buffer. During a multiplexor channel operation, data is read from the receiver data buffer and placed into memory whenever REQ becomes a one and ENABLE is set. REQ is then cleared to enable another transfer unless channel overflow occurs. In that case, UNUSUAL and LOW are set and REQ is not cleared. The MODE and TRANSMITTER OVERFLOW bits of the SR may be sensed to determine the status of the data transfer.

3.2 Maintenance Instructions

There are no special maintenance instructions.

3.3 Data Formats

Each byte sent by the transmitter is received in unaltered form. The format of the data depends upon the application.

Since each transfer sequence will bepreceeded by the transmission of a command list, a format should be designed for the particular application. The minimum information contained in the command list is the number of bytes about to be transferred.

3.5 Operator Controls

There are no operator controls.

3.6 Status Register

The Status Registers for the Bus to Bus Adapter appear as:

	8	9	10	11	12	13	14	15
Receiver	X MTR OVER-FLOW	MODE	UNUSUAL	0	REQ	X	LOW	ENABLE

(transfer occurs when data buffers are transferred)

	8	9	10	11	12	13	14	15
Transmitter	X	MODE	UNUSUAL	1	REQ	BUSY	LOW	ENABLE

(X indicates permanently zero)

The function of these bits is described in Section 3.1.

3.7 Programming Examples

Either the receiver or the transmitter or both may operate in the programmed transfer mode or in the multiplexor channel mode. In either case, the receiver should be initialized with its REQ bit cleared so that the first data byte may be transferred to it.

The examples presented will be for multiplexor channel operation of both the transmitter and the receiver.

3.7.1 Transmitter Routine

Assume that the main operating program develops a command list of the form:

```
LIST:  Byte Pointer To Data Block
       -Number Of Bytes Of Data
       3 Control Bytes
```

The transmitter routine will transfer all but the first 2 bytes of this command list (i.e. it will not transfer the position of the data block) to the receiver. After the command list has been received, it will transfer the data block.

In the following routine, the register labeled LOCK is used to lock the transmitter service routine. A zero value in LOCK indicates that the transmitter is still busy. A non-zero value indicates that the transmitter routines are free.

```
; INITIALIZE  LOCK

            CLR    LOCK                    ; CLEAR LOCK

            COM    LOCK                    ; SET LOCK TO -1

            -  -  -  -

; TRANSMITTER  ROUTINE
; CALLING SEQUENCE
;           BAL    TRNSMIT
;           POINTER TO COMMAND LIST
;           RETURN WITH TRANSMISSION STARTED
```

```
TRNSMT:     TST          LOCK         ; TEST LOCK FOR DONE
            BZ           TRNSMT       ; IF ZERO - STILL BUSY
            STA    3,    LOCK         ; SAVE ACCUMULATOR 3
            LDA    3,    @(2)         ; GET POINTER TO DATA BLOCK
            STA    3,    SDATA        ; SAVE POINTER
            LDA    3,    (2)          ; GET POINTER TO COMMAND LIST
            LDA    3,    1(3)         ; GET NUMBER OF DATA BYTES
                                      ; FROM COMMAND LIST
            STA    3,    SSIZE        ; SAVE SIZE OF DATA LIST
            LDA    3,    (2)          ; GET POINTER TO COMMAND LIST
            I            3            ; TURN IT INTO A BYTE
            RAL          3            ;
            AND    3,    [177776]
            STA    3,    XMTINT+1     ; STORE BYTE POINTER TO COMMAND
                                      ; LIST IN CHANNEL
            LDA    3,    [-5 ]        ; 5 COMMAND BYTES
            STA    3,    XMTINT       ; STORE IN CHANNEL
            LDA    3,    LOCK         ; RESTORE ACCUMULATOR 3
            CLR          LOCK         ; MAKE ROUTINE BUSY
            IOC    XMT,  [131]        ; SET MODE, ENABLE, REQ CLEAR
                                      ; OTHER BITS
            B            1(2)         ; EXIT FROM ROUTINE
LOCK:       0                         ; ROUTINE LOCK
                                      ; 0  -  BUSY
                                      ; -1 -  FREE
```

```
; SERVICE ROUTINE FOR TRANSMITTER INTERRUPT
; ENTER HERE WHEN CHANNEL OVERFLOWS
XMTSER:     IOT  XMT,  [100]        ; TEST MODE BIT
            BZ         XMTDUN       ; IF ZERO - DATA TRANSFER
                                    ; COMPLETE
            LDA  3,    SDATA        ; MODE =1 - COMMAND HAS BEEN
            STA  3,    XMTINT+1     ; SENT, NOW SEND DATA
            LDA  3,    SSIZE
            STA  3,    XMTINT       ; SET UP CHANNEL
            IOC  XMT,  [31]         ; SET REQ, ENABLE CLEAR REST
            PSD                     ; DISMIS
```

3.7.2 Receiver Routine

The normal or initialized state of the receiver is with its channel locations set up to receive 5 command bytes into its command list buffer. After the command list has been received, the receiver service routine will call a subroutine to allocate memory space for the following data. If space is available for the data, the channel will be set-up by the allocation routine which will then return control to the calling location +1. If memory space is not available, control will not return until it has been made available.

```
; INITIALIZE RECEIVER INPUT

INIT:   STA  3,   SAVE          ; SAVE AN ACCUMULATOR
        LDA  3,   [-5]          ; GET BYTE COUNT
        STA  3,   RCVINT        ; STORE IN CHANNEL
        LDA  3,   [2*COMLST]    ; GET POINTER TO COMMAND LIST
        STA  3,   RCVINT+1      ; STORE IN CHANNEL
        LDA  3,   SAVE          ; RESTORE ACCUMULATOR
        IOC  RCV, [1]           ; SET ENABLE, CLEAR OTHER BITS TO
                                ; ENABLE INPUT
        B         (2)           ; EXIT FROM SUBROUTINE
SAVE:   0                       ; TEMPORARY STORAGE
; SERVICE ROUTINE FOR RECEIVER INTERRUPT
; ENTER HERE WHEN CHANNEL OVERFLOWS
RCVSER:IOT  RCV,  [200]         ; TEST TRANSMITTER OVERFLOW
                                ; BIT, IF SET OK
        BZ        ERROR         ; IF NOT SET - DATA WAS LOST
        IOT  RCV, [100]         ; NOW - TEST MODE BIT
```

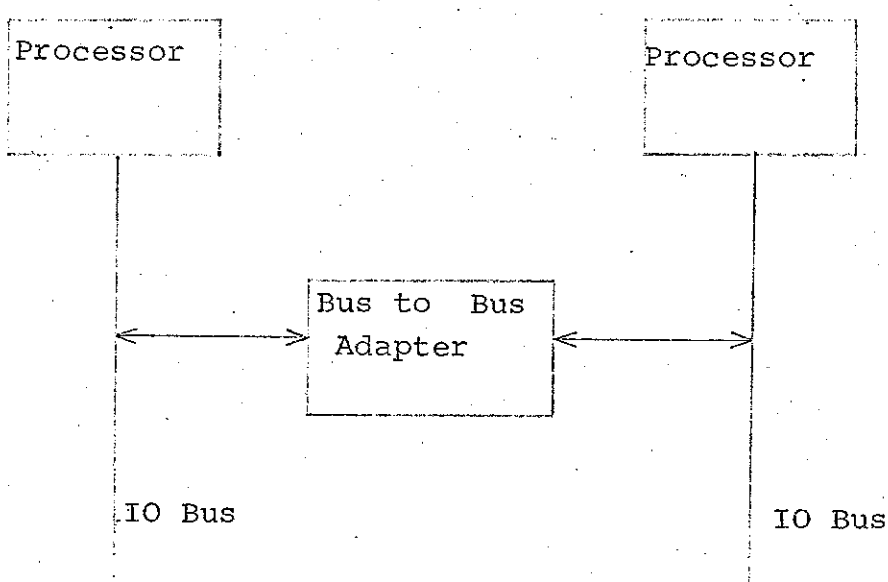
```
        BZ      RCVDUN          ; IF ZERO, END OF DATA
                                ; END OF COMMAND LIST
                                ; CALL ROUTINE TO SET UP
        BAL     ALLOC           ; CHANNEL FOR DATA TRANSFER
                                ; SINCE REQ IS SET, TRANSFER
                                ; WONT BEGIN UNTIL IT IS CLEARED
                                ; RETURN WITH CHANNEL
        IOC  RCV, [1]           ; SET UP - TURN ON ENABLE
                                ; CLEAR REST OF BITS
        PSD                     ; DISMIS
; COME HERE AT END OF DATA TRANSFER
RCVDUN:BAL      INIT            ; RE-INITIALIZE CHANNEL
        PSD                     ; DISMIS

; RECEIVER INTERRUPT LOCATIONS
LOC  RCVINT
        BLOCK 2                 ; RESERVE TWO WORDS FOR HIGH
                                ; INTERRUPT CHANNEL
        B       RCVSER          ; LOW INTERRUPT BRANCH TO
                                ; SERVICE ROUTINE
```

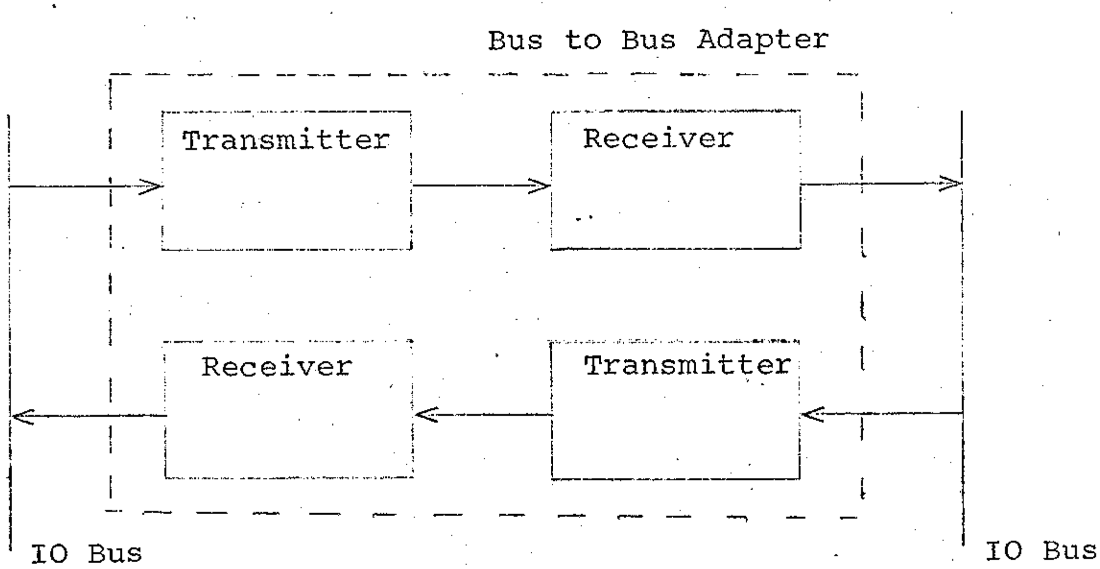
Since data transfers to the receiver may be stopped by leaving REQ set, the data may be read in many short records.

Appendix

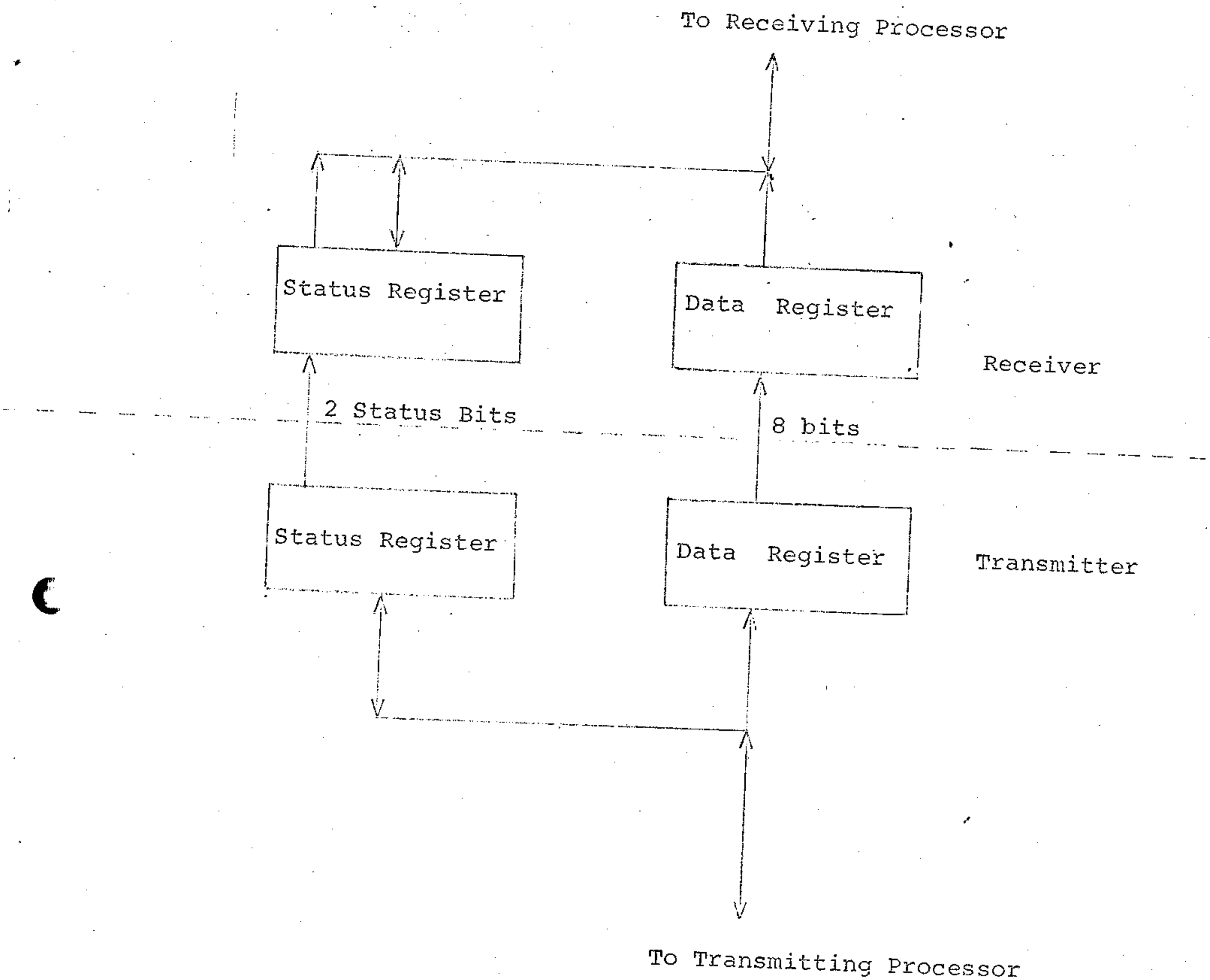
1. General Organization



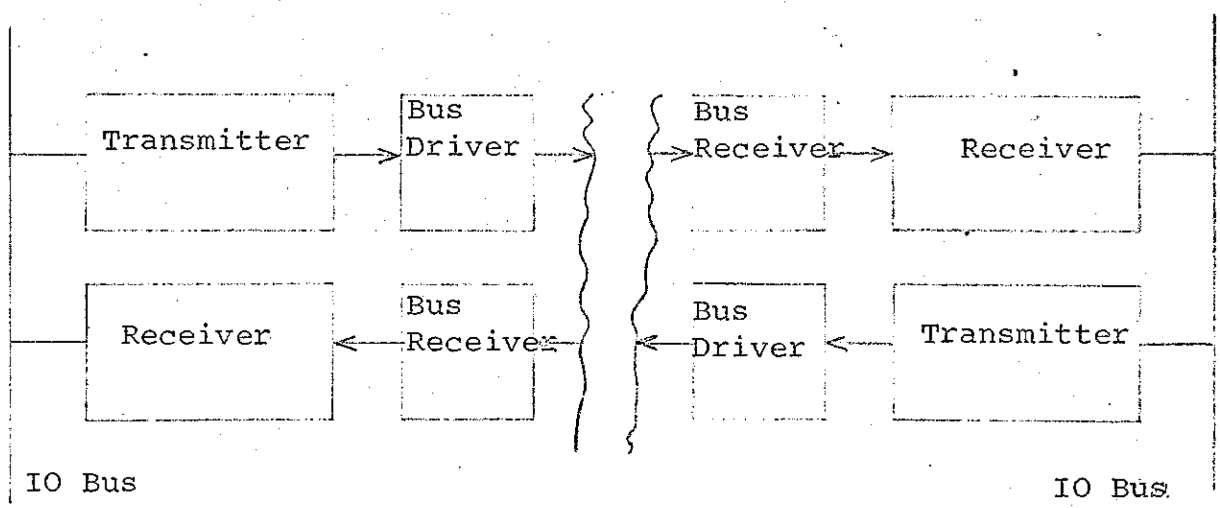
2. Detailed Organization



3. Single Pair

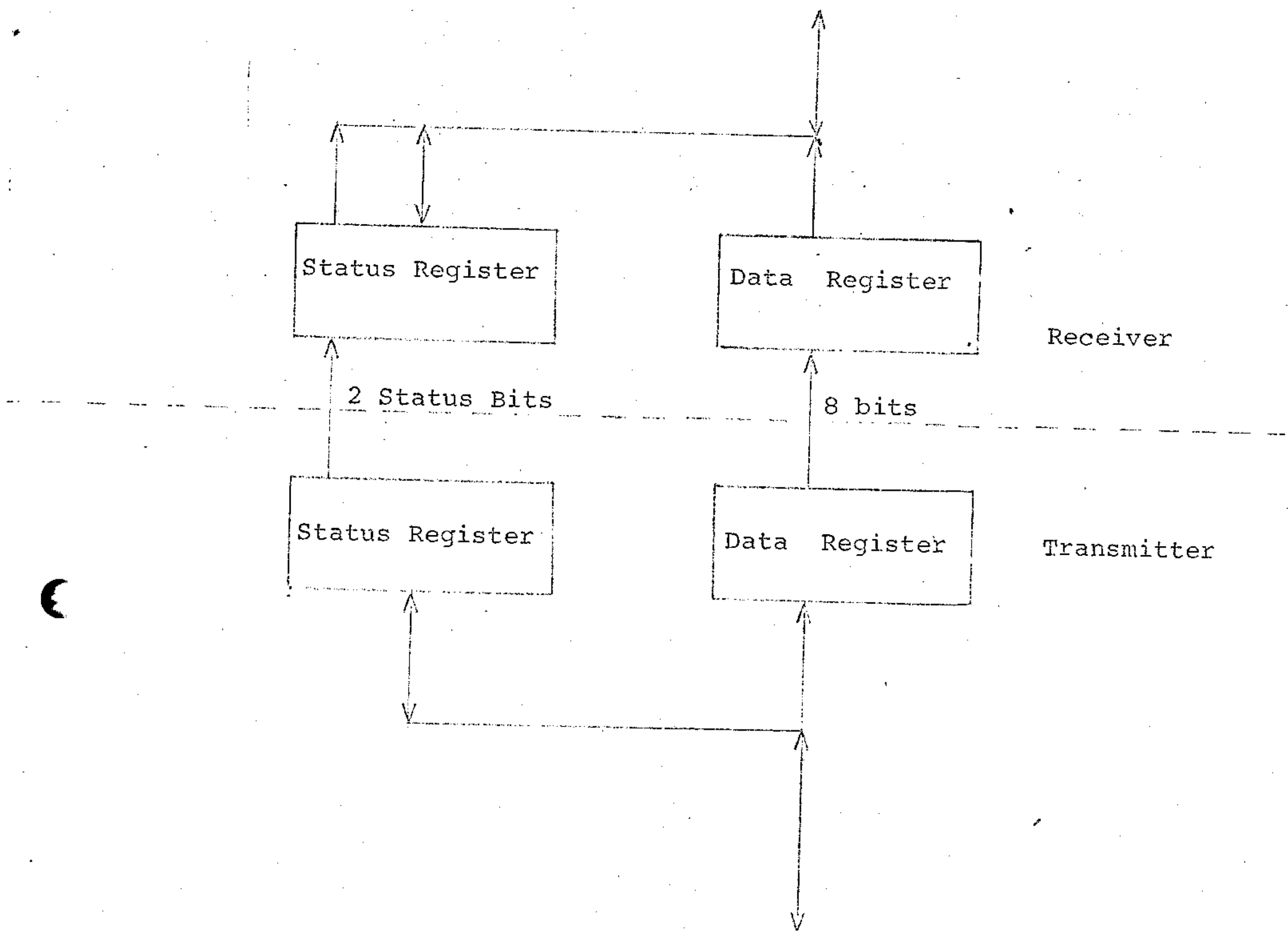


4. Long Distance



3. Single Pair

To Receiving Processor



To Transmitting Processor