Published: 06/14/68
(Supersedes: BF.5.05, 01/10/68)

Identification

The PRT202 Device Control Module
M. R. Bianchi

Purpose

This paper describes the PRT202 Device Control Module (DCM) and how it is used to drive the PRT202 printer. A detailed knowledge of the GIOC, the GIOC Interface Module (GIM) and of the PRT202 printer is a requisite in order to understand this document.

Introduction

The role of the PRT202 Device Control Module (henceforth, the DCM) is to drive efficiently the General Electric PRT202 printer. The DCM accepts, usually from the PRT202 Device Strategy Module (DSM), physical record-oriented calls specifying six-bit data elements suitable for printing on the PRT202 printer and issues calls to the GIM to have them printed. Each physical record of data received by the DCM is assumed to be one print line with the appropriate slew characters appearing at the end of the line. The DCM does not examine the data being written and all the I/O errors are returned to the caller by the standard Multics IOS status mechanism.

While the DCM is designed to cope with the PRT202 printer, it is also forced to deal with the GIOC through the GIM. Certain peculiarities of its construction are due to its GIM interface. Restrictions imposed by the GIM are mentioned in the body of this documentation, as are restrictions enforced by the hardware being used. Efficiency considerations are briefly treated.

Strategy

The PRT202 printer is capable of printing lines at a rate between 600 and 1200 lines per minute, depending upon the composition of the lines being printed. The printer drum has a thirty-four character subset of the ninety-four ascii graphics duplicated so that lines containing only these thirty-four characters can be printed at the higher rate, while the introduction of any other character causes the rate to drop to 600 lpm. At the lower rate, the printer

prints one line every one hundred milleseconds; it is anticipated that the data offered to the printer will normally enforce this lower rate. Based on this assumption, the DSM-DCM strategy is to queue enough lines of data in the GIOC so that the next output call could be processed and added to the hardware queue before the GIOC has terminated its activity. In this way, the printer might be kept printing continuously at the rate allowed by the data, provided that the data processing takes place at a rate below one hundred milleseconds per line and that the frequency of calls to the DCM is correlated with the number of lines in the hardware queue.

The data control word list

In order to implement the PRT202 driving strategy, the GIOC is programmed to operate from a "circular" list of data control words (DCW's) which is kept active as long as sufficient data are supplied. Only one list of DCW's is used by the DCM. The list consists of a set of pairs of DCW's, each of which is made up of an Instruction DCW (IDCW) and a Data DCW (DDCW). The final DCW in the list is a transfer to the first DCW of the list. Each IDCW-DDCW pair can cause a single line to be printed. At any given time, the list will contain precisely one IDCW with the "end data transfer" bit on (to cause the GIOC to terminate). Certain other IDCW's may have the "external signal" bit on (to cause the GIOC to create an external signal interrupt). Provided that the list is active, a new writerec call is processed by overwriting the terminate IDCW, changing the following DDCW's to refer to the lines given in the writerec call, and placing a new terminate IDCW after the last DDCW. If the bottom of the list is reached by the GIOC, the transfer DCW will cause it to begin again at the top of the list where new DDCW's have been added. A block of lines specified in a writerec call will be added in bursts of lines equal to the number of free IDCW-DDCW pairs in the list by hcs_$list_change calls to the GIM. If there are too many records specified to fit in the currently available DDCW slots, then as many lines as available DDCW slots are added to the list, and the extra lines are queued in a DCM buffer until more DDCW slots become free.

Status and Interrupts

In the scheme described above, no termination status is stored by the GIOC in the course of normal operation. Each time the DCM regains control, it can ascertain the current position of the GIOC in the DCW list and hence the number of free IDCW-DDCW pairs in the list. Since

a list may not be extended beyond its original length by calls to the GIM, it is necessary to overwrite the first pair in the list after the last one has been filled in; but it is not safe to do so until the GIOC has finished processing the first pair. The knowledge of the GIOC's current position in the list, combined with a record of which pairs have not been processed, provides safety in overwriting parts of the list.

It is also necessary that the caller of the DCM receive information concerning the transactions which his calls have caused. A typical PRT202 DSM strategy, for example, might require that it be awakened by a hardware interrupt in order to continue making calls to the DCM. To aid in this sort of strategy, the DCM places external signal IDCW's in its list everytime that it overwrites a terminate IDCW or with every tenth IDCW that it adds to the list at one time. The external signal IDCW's do not stop the data transmission between the GIOC and the PRT202 printer but force the GIOC to store a status word and to cause an interrupt without causing a termination on the channel. The interrupt will awaken the DCM which causes ultimately the DSM to be awakened by updating the transaction status.

Call Interfaces

The IOS outer calls presently accepted by the DCM are: attach, detach, writerec, abort, order and upstate

To facilitate design, implementation and future amelioration, the DCM is divided into four logical modules:

1. The Initializer
2. The Initiater
3. The Terminater
4. The Data Bases Maintainer

The initializer handles the attach IOS outer call. The initiater handles the writerec and order IOS outer calls. The terminater handles the upstate, abort and detach IOS outer calls. Finally the data bases maintainer allocates, maintains and deallocates the data bases which are utilized by the DCM to represent the writerec and order calls.

An explanation of the design and implementation of the DCM's four logical modules follows.

The Data Bases Maintainer

Each of the other three DCM logical modules utilizes a per-Ioname Segment (IS) and a Transaction Block Segment (TBS) which are created by the I/O Switching Complex (BF.2.10) to preserve their global nonautomatic data and their per-transaction nonautomatic data.

The DCM global nonautomatic data are kept in one Per-Ioname Base Extension (PIBE) (BF.2.20). The DCM PIBE has the following declaration:

```
dcl  1 pibe based (p),
       2 pibe_chain,                      /*standard pibe chaining*/
          3 next_pibe bit (18),           /*relative pointer to
                                            next pibe*/
          3 pibe_length bit (18),         /*relative ptr to last
                                            item in this structure*/
       2 dcw_list_size fixed bin (17),
                                          /*size of the list used
                                            by the gim*/
       2 dcw_list_activity fixed bin (17),
                                          /*activity status of the
                                            dcw list  = 0 not active
                                                      = 1 active */
       2 last_terminate fixed bin (17),
                                          /*location of terminate
                                            in dcw list*/
       2 unprocessed_queued_tbindex bit (18),
                                          /*first unprocessed
                                            queued tb*/
       2 unprocessed_tbe_chain,           /*chain of queued tbes*/
          3 first chained_tbe ptr,
          3 last_chained_tbe ptr,
       2 device_index fixed bin (17),/*device index returned
                                            by the gim*/
       2 data_packing_mode fixed bin (17),
       2 special_interrupt_switch,
          3 switch bit (1),
          3 tbe_pointer ptr,
       2 prt202_error fixed bin (17),/*error switch*/
       2 rqs_type fixed bin (17),
       2 free_dcw fixed bin (17),     /*number of free dcws in
                                            the list*/
       2 first_dcw fixed bin (17),
       2 last_dcw fixed bin (17),
       2 total_dcw fixed bin (17),
       2 abort_condition fixed bin (17),
       2 dcw_list (dcw_list_size) ptr,
       2 last_item fixed bin (17);    /*last item in pibe*/
```

The DCM preserves the nonautomatic data associated with each transaction in one individual Transaction Block Extension (TBE) (BF.2.20). The TBE has the following declaration:

```
dcl  1 tbe based (p),
       2 tbe_chain,                    /*standard tbe
                                         chaining*/
           3 next_tbe bit (18),        /*relative pointer to
                                         next tbe*/
           3 tbe_length bit (18),      /*relative ptr to last
                                         item in this structure*/
       2 next_tbe ptr,
       2 status_string bit (144),      /*transaction status*/
       2 tbe_type fixed bin (17),      /*specifies function
                                         of tbe*/
       2 tb_index bit (18),            /*tb associated with
                                         present tbe*/
       2 count fixed bin (17),         /*number of physical
                                         records*/
       2 count2 fixed bin (17),        /*next unprocessed
                                         physical record*/
       2 count3 fixed bin (17),        /*last processed
                                         physical record*/
       2 write_record (count),         /*description of the
                                         physical records*/
           3 workspace ptr,            /*pointer to each
                                         physical record*/
           3 nelem fixed bin (17),     /*number of elements in
                                         each physical record*/
           3 nelemt ptr,               /*pointer to caller
                                         argument nelemt array,
                                         it is to indicate the
                                         number of elements
                                         successfully printed
                                         for each physical
                                         record*/
           3 offset fixed bin (17),    /*specifies the first
                                         element in each physical
                                         record*/
       2 last_item fixed bin (17);     /*last item in tbe*/
```

The data bases maintainer is made up of two procedures:

1. The TB housekeeper
2. The TBE housekeeper

The TB housekeeper has two entries

1. Allocate
2. Deallocate

Everytime that the PRT202 DCM receives control through the I/O Switching Complex, a Transaction Block (TB) is allocated by the Transaction Block Maintainer (TBM) automatically (BF.2.20).

When the DCM initiater receives control from the I/O Switching Complex after a PRT202 DSM writerec or order call, it is necessary to queue and preserve the TB until the transaction specified by the outer call has been initiated and terminated. This is done by calling the allocate entry of the TB housekeeper. The TB housekeeper queues the new TB to the PIBE chain of unprocessed TB's and preserves it by a call to tbm$set_hold of the TBM which sets hold bit 2 of the TB and therefore prevents its deallocation. When the requested transaction has been completed either successfully or unsuccessfully, the DCM must inform the requester of the transaction of the termination and allow for the release of the TB. These actions are done by a call to the deallocate entry of the TB housekeeper. The TB housekeeper posts the transaction TB with the termination status by a call to tbm$set_status of the TBM. However, if the status of the terminated transaction indicates that the transaction was aborted -status bit 14 on- the TB housekeeper acts differently. The present TB and all the queued TB's, which are fetched by a call to tbm$get_chain of the TBM, are posted with the identical termination by the same call to the TBM as before, and allowed to be removed by a call to tbm$delete_chain of the TBM.

Since the transactions requested by the IOS outer call writerec and order might not be initiated immediately (the DCW list might be full) and have, also, arguments, which must be set upon termination of the transactions (such as the number of characters which have been printed), it is therefore necessary to preserve the arguments passed with these two IOS outer calls until the transactions are terminated. This is accomplished by storing the arguments in a TBE. A TBE is allocated by a call to the TBE housekeeper.

The TBE housekeeper allocates a TBE in the DCM IS, and sets a pointer to it in the TB representing the transaction by a call to tbm$set_tbe of the TBM. It then queues the TBE to the PIBE chain of unprocessed TBE's and initializes it with such parameters as the index of the TB and a termination status frame indicating successful termination (status bits 4 and 5 set). By storing the TB index in the TBE, the DCM is able to identify, without calling the TBM, the TB associated with any transaction. Figure 1 shows the relationships of the data bases used by the DCM.

The Initializer

The initializer gets control when the PRT202 DSM issues the following IOS outer call:

```
call attach (ioname1, typename, ioname2, mode, status,
     pibptr);
```

where

ioname1 is the attached ioname,

typename is the entry name of the type directory in the Registry File Directory,

ioname2 is the directory name of the Registry File Directory,

mode is the mode requested by the PRT202 DSM,

status is the returned status string,

pibptr is a pointer to the base of the Per-Ioname Base (PIB) which is provided by the I/O switch.

In response to the attach call, the initializer takes the following steps:

1. calls disp$get_hardware (BF.2.25) in order to discover the name of the hardware event channel for the PRT202 printer. This event channel is the one that will be signaled by the GIM whenever a hardware interrupt is detected for the printer. The dispatcher (BF.2.25) will also indicate if the present I/O path is the only one for this device. If the I/O path is not the only one, it is considered to be an error, in which case the initializer sets status bits 5, 6, 8 and 91, and returns to the caller.

2. calls rfm$get_devices (The Registry File Maintainer, BF.2.22) to get the resource name. If an error condition is returned by the Registry File Maintainer, the initializer passes the error to the caller by setting status bits 5, 6, 8 and 92, and returns.

3. calls mode$bset (Mode Handler, BF.2.27) to process the mode that was passed with the attach call. If the Mode Handler returns an error condition, the initializer sets status bits 5, 6, 8 and 93, and returns to the caller.

4. initializes the following elements of the PIB:

   1. ioname1
   2. typename
   3. ioname2
   4. bmode

   with respectively the ioname1, typename and ioname2 arguments of the attach call and the bmode bit string returned by the Mode handler. The "element_size" and "bound_bit" elements of the PIB are set to 6 and 131072.

5. proceeds to allocate the DCM PIBE in the IS and then initializes it.

6. gains access rights to a GIOC channel by calling the GIM with the call hcs_$assign. If the GIM returns an error code, the initializer sets status bits 5, 6, 8 and 94, and returns.

7. the initializer terminates by setting status bits 4 and 5, which indicates that the attach call was successful, and returns.

One can notice the fact that the initializer does not test the PRT202 printer to find out if it is in a ready status. The PRT202 DSM can, if it desires, find out the PRT202 printer status by an order call (described later) to reset the PRT202 printer status.

The Initiater

The DCM initiater handles the two following PRT202 DSM outer calls:

1. call writerec (ioname1, rec-count, buffer, offset, nelem, nelemt, status, pibptr);

where

ioname1 is the attached ioname,

rec-count indicates the number of physical records which the writerec call represents,

buffer is an array of pointers to the corresponding physical records,

offset is an array of values indicating the starting offset elements in each physical record,

nelem is an array of values indicating the number of elements in each physical record,

nelemt is an array of values which will specify to the PRT202 DSM how many elements were actually printed with each physical record,

status returned status string,

pibptr pointer to the base of the PIB which is provided by the I/O Switch.

2. call order (ioname1, request, argptr1, argptr2, status, pibptr);

where

ioname1 same as above,

request indicates the ordered request,

argptr1 not used,

argptr2 not used,

status same as above

pibptr same as above

Before reading the strategy of the initiater it is important to note that:

1. The PRT202 DCM employs the "print_in_edited_mode_no_slew instruction" command to drive the PRT202 printer. It is therefore the responsibility of the PRT202 DSM to insert the proper escape and slew characters at the end of each physical record.

2. The PRT202 DCM assumes that the physical records are packed six characters per word beginning at the left-most character. Therefore the PRT202 DSM has the responsibility to make sure that the address of the beginning of each physical record, which is formed by adding the offset divided by six to the buffer pointer, falls on a full word boundary.

The DCM initiater is made up of four modules

1. The prt202_dcm_writerec
2. The prt202_dcm_order
3. The prt202_dcm_initiater
4. The prt202_dcm_rqs

The prt202_dcm_writerec handles the writerec call, the prt202_dcm_order handles the order call, the prt202_dcm_initiater initiates the queued writerec calls by calls to the GIM and the prt202_dcm_rqs initiates the "request status" and "reset status" requested by the order call.

In response to a writerec call, the initiater (prt202_dcm_writerec) takes the following steps:

1. tests the DCM PIBE error switch. If it is on, indicating that either a PRT202 I/O error has occured previously or an abort call was received, the initiater sets status bits 5, 8, 14 and 96 on and returns.
2. tests the value of the rec_count argument. If it is equal to zero, the initiater sets status bits 5, 8, 14 and 97 and returns.
3. calls the TB housekeeper to queue and initialize the TB representing the writerec call.
4. calls the TBE housekeeper to allocate, initialize and queue a TBE. If the TBE housekeeper returns an error condition caused by an area overflow during the allocation of the TBE, the initiater sets status bits 5, 8, 14, 96 and 97, and returns.
5. sets status bit 3, which indicates a successful physical initiation, and calls the prt202_dcm_initiater module of the initiater. Upon return from it, the initiater returns to the PRT202 DSM.
6. The initiater (prt202_dcm_initiater) finds out if there are some free IDCW-DDCW pairs. (The initiater keeps an IDCW-DDCW pair counter in the DCM PIBE. This counter is decremented by the number of IDCW-DDCW pairs which have been processed by the GIOC. The number

of IDCW-DDCW pairs, which have been processed by the GIOC, is derived by the DCM terminater after every termination and external signal interrupt. It is why, in order to keep the number of free IDCW-DDCW pairs greater than zero, the initiater inserts an IDCW-DDCW pair with the external signal bit on when more than ten IDCW-DDCW pairs are added to the DCW list). If none are available, it returns to the caller.

7. gets the first TBE from the queue of the unprocessed TBE's. If the queue is empty, it returns.

8. calls the GIM with the hcs_$list_size call if no DCW list has already been assigned in the contiguous wired-down core of the GIM. This step is necessary for the DCM does take advantage of lulls in activity on the PRT202 printer to release the wired-down DCW list and buffers used by the GIM to drive the GIOC. Presently the DCM uses a DCW list size of 62; this size can be increased or decreased by recompiling the initializer module.

9. initializes the free IDCW-DDCW pairs by using the writerec call arguments stored in the TBE and by calling the GIM with either one or two calls hcs_$list_change depending on whether or not the DCM has reached the end of the DCW list.

10. tests the PIBE activity switch. If it is on indicating that the DCW list is being processed by the GIOC go to step 6. If it is off, the initiater calls the GIM with the call hcs_$list_connect to request it to activate the DCW list, and goes to step 6.

This strategy assumes that the DCW list is active until a termination interrupt occurs. It is also noted that the initiater does not use the hcs_$get_cur_status call to the GIM before making changes to the DCW list. This is not necessary since the DCM knows at any time how many IDCW-DDCW pairs are available and which they are (refer to step 6). Furthermore the initiater does not issue a hcs_$get_status call to the GIM after the DCW list has been changed to find out if the DCW list patch was successful. It assumes that it was. If the patch was not successful, and instead a termination interrupt occured before the patch took, the DCM terminater would regain control through an upstate call. The terminater then issues a hcs_$get_status

call to get the channel status. If the status defines a termination interrupt without any error, it will call the initiater which will change the DCW list if it is necessary and call the GIM to connect it.

Finally this initiater strategy will be more efficient if the PRT202 DSM uses a writerec call which defines several physical records.

Presently the PRT202 DSM can use the order call to request the following actions:

1. reset the I/O error switch in the PIBE (order 1)

2. to be signaled when a special interrupt is generated by the PRT202 printer (order 2)

3. request subsystem PRT202 printer status (order 3)

4. reset subsystem PRT202 printer status (order 4)

The order requested is indicated by the request argument of the order call. Orders 1, 2, 3 and 4 are defined respectively by the character string of the "request" argument: 0, 1, 2, 3 .

Order 1 is used by the PRT202 DSM to inform the DCM either that all the PRT202 printer I/O errors have been corrected or that it received a restart call from the dispatcher following a quit condition, and that it is ready to resume its writerec calls to the DCM. The I/O error switch is set by the DCM when either the PRT202 printer terminates with an I/O error or it receives an abort call from the dispatcher after a quit interrupt.

Order 2 allows the PRT202 DSM to be able to receive the operator communication entered through the actuation of any of the eight input control switches located on the PRT202 printer and which create a special interrupt. Order 2 can be requested only if the I/O error switch is on.

Orders 3 and 4 allow the PRT202 DSM to be able to request or reset the subsystem PRT202 printer status. These last two orders only can be used when the DCW list is inactive.

Upon receiving an order call the initiater (prt202_dcm_order) performs the following steps:

1. if an order call is to reset the I/O error switch in the PIBE, the initiater resets it, sets status bits 4 and 5, and returns.

2. if the order call is for an order 2 and the I/O error switch is not on, the initiater sets status bits 5, 8, 14 and 99 and returns. Otherwise two possible courses of action exist. If a special interrupt occured before the order call, the initiater sets status bits 4 and 5, and returns; if there is no waiting special interrupt, the initiater calls the TBE housekeeper to allocate a TBE and queues it until a special interrupt occurs. Then status bit 3 is set and control returns to the caller.

3. If an order 3 or 4 is requested, the initiater sets status bits 5, 8, 14 and 100, if the I/O error switch is not on. Otherwise, it calls first the TBE housekeeper and then the prt202_dcm_rqs procedure gains control. This procedure builds the proper IDCW for the request or reset subsystem status, changes the DCW list and connects it. Upon return from the prt202_dcm_rqs procedure, the initiater sets status bit 3 on and returns to the caller.

4. If the initiater receives an order call for a request which is not presently implemented, it sets status bit 5, 8, 14 and 98 and returns.

The Terminater

The terminater handles the following three IOS outer calls:

1. call upstate (ioname1, status, pibptr);

where

ioname1 is the attached ioname,

status returned status string,

pibptr pointer to the base of the PIB.

2. call abort (ioname1, oldstatus, status, pibptr),

where

ioname1 as above,

oldstatus identifies the transaction which must be aborted,

status as above,

pibptr as above.

3. call detach (ioname1, ioname2, disposal, status, pibptr);

where

ioname1 as above,

ioname2 is the directory name of the Registry File Directory,

disposal is not used,

status as above,

pibptr as above.

The terminater receives an upstate call from the dispatcher whenever an interrupt occurs on the channel connected to the PRT202 printer. It receives an abort call from the dispatcher whenever a quit condition occurs. Finally the terminater receives a detach call either from the PRT202 DSM when all its user I/O's are terminated or from the dispatcher after a logout occurs without a DSM detach call.

The terminater is responsible for returning the hardware status to the DSM by updating the TB's representing the I/O transactions terminated or in error; and also for giving control to the initiater so that queued TB's can be processed. The terminater handles three types of I/O interrupts:

1. Termination
2. External signal
3. Special interrupt

Upon receiving an upstate call the terminater performs the following steps:

1. Calls the hcs_$get_status entry of the GIM in order to get a maximum of three hardware status frames and analyses each status frame.

2. If the status frame defines a termination interrupt, two possibilities exist: the termination status indicates that the channel/peripheral subsystem is ready and therefore the I/O terminated without any error, or it indicates that the I/O terminated with an error.

3. If there is an I/O error, the I/O error switch is set and the status string of the TB representing the writerec call has bits 5, 6, 9 and 14 set, and bits 115 to 126 are set to the adapter status. Also the "nelemt" argument of any physical record which was successfully printed is set to the number of elements printed. Go to step 8.

4. If there is no I/O error, the associated TB status string bits 4 and 5 are set as are the "nelemt" arguments. Go to step 8.

5. If the status frame defines an external interrupt go to step 4.

6. If the status frame defines a special interrupt, the terminater either posts bits 4 and 5 of the TB waiting for the special interrupt or if there is no TB waiting it sets a switch in the PIBE indicating that a special interrupt has occured. Go to step 8.

7. If the termination interrupt was caused by the completion of a request or reset subsystem status, the terminater sets bits 115 to 126 of the associated TB equal to the adapter status.

8. The terminater calls the TB housekeeper so that all terminated -either in error or not- transaction TB's are posted and removed from the main chain. All the previous steps are repeated until all the hardware status frames have been processed.

9. If the I/O error or abort switch is set go to step 10, otherwise the terminater calls the initiater so that queued TBE's may be initiated and/or the DCW list connected.

10. Upon return, the terminater will first release the GIM wired-down buffers by the call hcs_$list_size to the GIM if the DCW list is inactive, and second will set the upstate call status bits 4 and 5 before returning to the caller.

In response to an abort call, the terminater performs the following actions:

1. sets the I/O error switch and the abort switch.

2. If the DCW list is inactive, go to step 4.

3. If the DCW list is active, the terminater calls the GIM with the call hcs_$list_connect to reset the channel active mode and therefore stop the channel.

4. sets status bits 4 and 5 and returns to the caller.

In response to a detach call, the terminater behaves as follows:

1. If the DCW list is active, the terminater sets status bits 5, 8, 14 and 101, and returns.

2. If the DCW list is inactive, the terminater calls the GIM with the call hcs_$unassign so that the GIM will relinquish the channel used by the PRT202 printer.

3. sets status bits 4 and 5, and returns to the caller.

PRT202 DCM Error Handling

The DCM does not try to correct any I/O errors, it just passes their description to the PRT202 DSM which can, then, take the appropriate corrective measures. The I/O errors pass to the PRT202 DSM can be divided into three logical groups:

1. Hardware

2. Abort

3. Operator

The hardware errors are the ones which are caused by the adapter or the PRT202 printer. An abort error is created when the DCM receives an abort call. An operator error is created when the operator presses either the manual halt or any of the input control switches. The PRT202 DSM can find out what type of I/O error has occured by analysing the status of the TB. If status bit 5 is on and bit 4 is off the PRT202 DSM knows that a hardware error has occured if status bit 9 is also on. If status bit 15 is set, it indicates that the I/O transaction was aborted because of an abort call. The only way that the DSM can tell if it is an operator generated error is by finding out if the hardware adapter status returned in

status bits 115 to 126 indicates a manual halt. When an I/O error occurs, the TB associated with the error and all the queued TB's are aborted (status bit 14 on). Also the DCM will not accept any more writerec calls until the I/O error is corrected by the PRT202 DSM. The PRT202 DSM informs the DCM that the I/O error is corrected by issuing an order call to reset the I/O error switch. The PRT202 DSM can find out the status of the PRT202 printer by issuing an order call to request or reset the PRT202 status. Table 1 shows the relationship between status bits 115 to 126 and the PRT202 printer status.

Table 1.

Relationship between TB status bits 115 to 126 and the PRT202 printer.

bits 115, 116 are ignored
bits 117 to 120 = PRT202 major status
bits 121 to 126 = PRT202 substatus

1. Meaning of different patterns after termination interrupt not following a special interrupt

major status	= 0000	channel/peripheral subsystem ready.
	= 0010	attention condition (see substatus)
	= 0011	data alert (see substatus)
	= 0101	command reject (see substatus)
	= 1001	peripheral absent or power off
	= 1010	parity error detected by channel

attention condition

substatus	= 000001	out of paper
	= 000010	manual halt
	= 000100	VFU tape alert
	= 001000	check (printer physical failures)

data alert

substatus	= 000001	transfer timing error
	= 001000	paper low
	= 010000	slew error

command reject

substatus	= 010000	slew error

2. Meaning of the different substatus patterns after a special interrupt (operator communication).

substatus	= 000000	normal (communication terminated)
	= 000001	print one line
	= 000010	forward space
	= 000011	forward to top of page
	= 000100	invalid line
	= 000101	reverse rewind
	= 000110	backspace
	= 000111	backspace to top of page

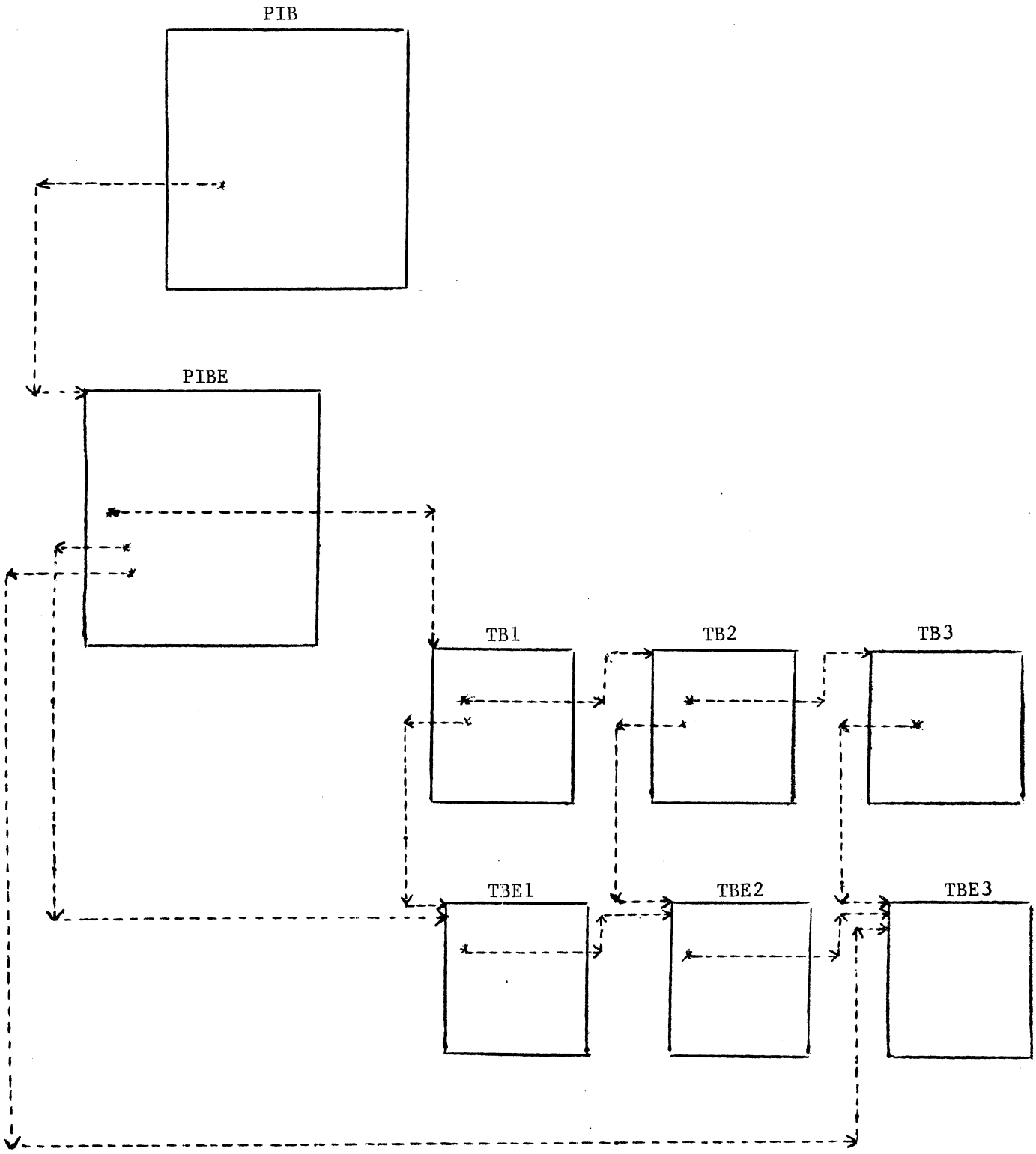

Figure 1. Relationships between the data bases used by the DCM.