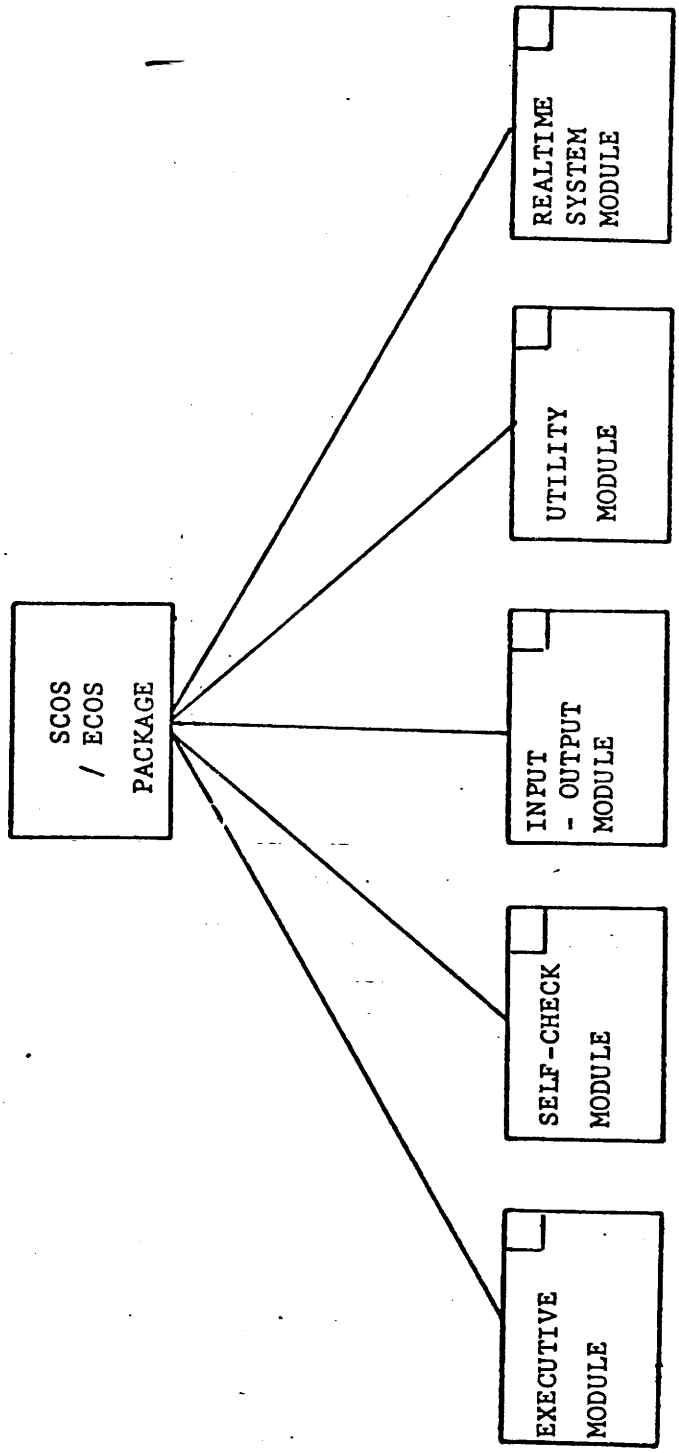


FIGURE 3 - SCOS/ECOS PACKAGE FUNCTION SET DIAGRAM

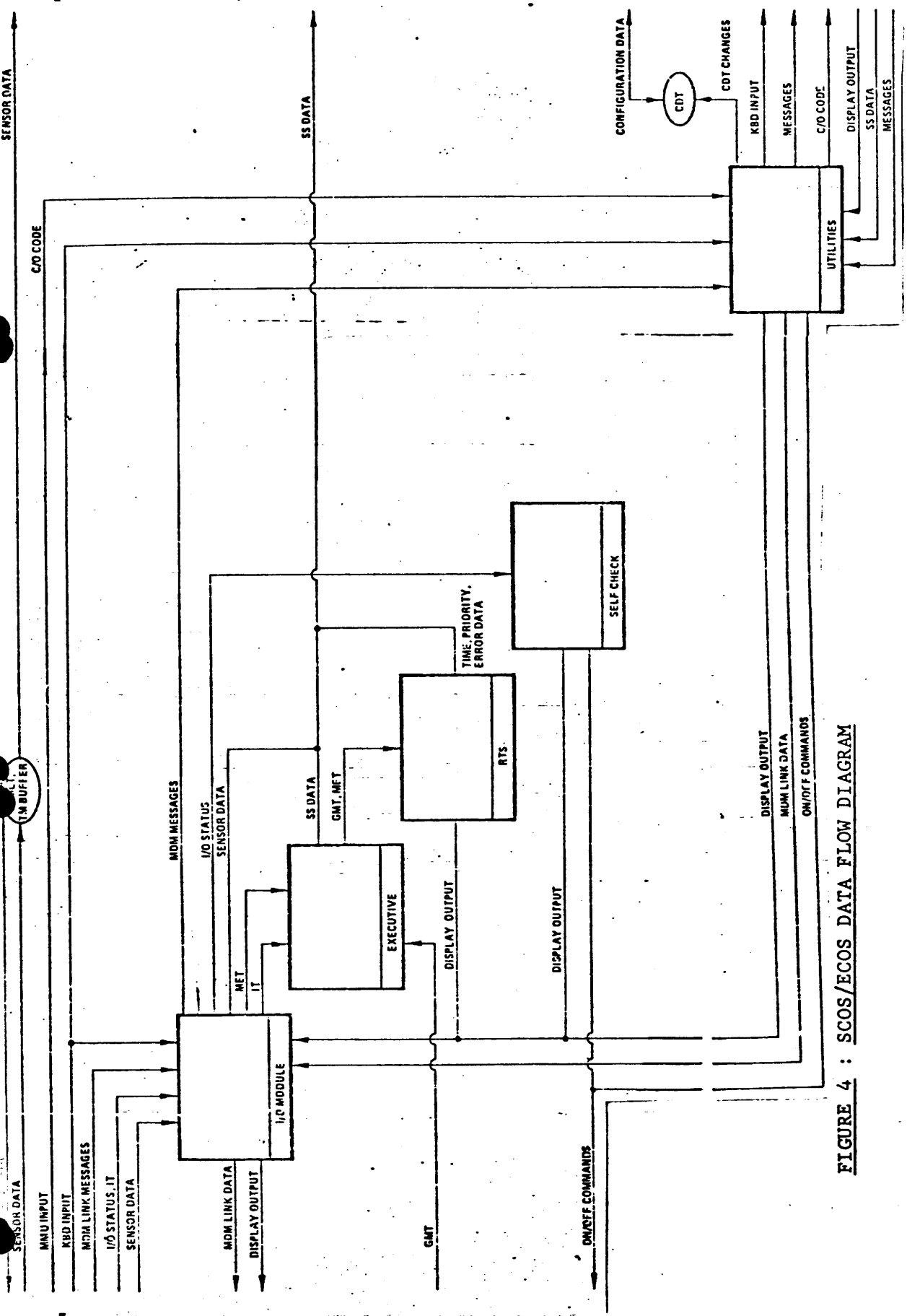


FIGURE 4 : SCOS/ECOS DATA FLOW DIAGRAM

1.4.4 - SCOS General features

Figure 3 gives the module decomposition of SCOS package. Figure 4 shows the main data flow between SCOS modules. The following section gives a brief overview over SCOS structure.

The Øs settles under base S and is made up of the following components :

a) ØS data

H/W to S/W interface memory locations

System tables

Communications vectors

Constants.

b) Supervisor sections

These sections are called by CSV instruction and perform the services offered by the ØS to minimize the effort of programming user programs.

c) Handlers

These modules control the physical I/O transfers to the peripherals connected to the Mitra 125.

Each handler is an "immediate task" connected to the interrupt level dedicated to that peripheral.

d) System tasks

d.1) ON/RESTART task

This level is triggered by an internal interrupt when the main power is turned on or when the main power comes back after a shut-down.

d.2) TIME MANAGEMENT task

The corresponding level is activated by the time coupler interrupt (every 10 ms). This task may manage different timings :

absolute time (GMT)

time - out (on I/O transfers)

delays (on program request).

d.3) Measurement point acquisition (GML)

d.4) Time Display task

d.5) IOU general interrupt task

d.6) See figure 5.

Systems tasks (belonging to the operating system) may not be independent. This situation will be reflected by a system generation process which is performed on the ground and depends on the CDMS configuration.

e) Internal failures handling

Several internal cases of errors are handled by SCOS :

- size overflow
- mode violation
- memory protection
- inexistant address
- memory parity error
- inexistant instruction

If the error does not prevent the operating system working, then the task where the error occured will be aborted.

If the error prevents the operating system working, then the system will go to a halt after saving necessary informations for checking.

More information will be found in the description of S:TRAP module (§ 4.3.3).

f) Application tasks

Each user application program is connected to a separate interrupt level.

The level 0 task is considered as a background task with the lowest priority and as a result, has to be handled in a slightly different way.

IT LEVEL	TYPE	125 MS	125 S	REF.	REMARKS
32	INT	T:AUTO	E	2.1.4	COMPUTER POWER-ON
31	|	E	T:ITOF		COMPUTER POWER-OFF
30		E	T:AUTO		
29		E			BUS EXTENSION
28	H/W	T:RTC	- id -	3.3.2	COMPUTER RTC
27	|	T:SAVE	- id -	2.1.4	IOU POWER-OFF
26	|	T:ITON	- id -	2.1.4	IOU POWER-ON
25	|	T:MTU	- id -	3.3.2	TIME COUPLER
24	|	T:IOU	- id -	3.1	GENERAL IOU INTERRUPT
23	|	EXP			RESERVED FOR EXP
22	|	"	DRI		"
21	|	"	LX		"
20	|	"	CR		"
19	|	E	M9T		ONLY THESE 2 PERIPHERALS CAN
18	|	E	KSR		BE CONNECTED TO 125 S
17	S/W	T:GML	S/W	3.2.2	RAU ACQUISITION
16	|	T:MDM	|	3.7.4	MDM PROCEDURE
15	|	T:KBD	|	3.4.3	KEYBOARD TASK
14	|	T:TIME	|	3.4.4	TIME DISPLAY
13	|	Application	12		
12	|	|	11		
11	|	|	10		
10	|	|	9		
9	|	|	8		
8	|	|	7		
7	|	|	6		
6	|	|	5		
5	|	|	4	PUP-ITRE	
4	|	|	3		
3	|	|	2		
2	|	Application	1		
1	S/W	T:SELF	|	4.3.4	SELF TEST TASK
0	S/W	|	|	2.3.2	BACKGROUND TASK

INT : internal interrupt

E : empty level = cannot be used

FIGURE 5 - INTERRUPT LEVEL ASSIGNMENTS

All tasks are independent from the operating system point of view :

- each task is compiled/linked separately

- each task is loaded without affecting the processing of the other running tasks.

There are different means of communication between tasks :

- with the operating system support :
  - . task activation (triggering IT level)
  - . event activation (processing an associated CB)

- using the common area :

This area is used by application tasks to pass data between themselves.

The common area is defined on a consensus basis and each task knows its structure (since the link-edit time).

The protection of this area can be achieved in two complementary ways :

- ajusting the length area for each task to the appropriate value (Z1)

- setting the protection bit in every memory location which is read only.

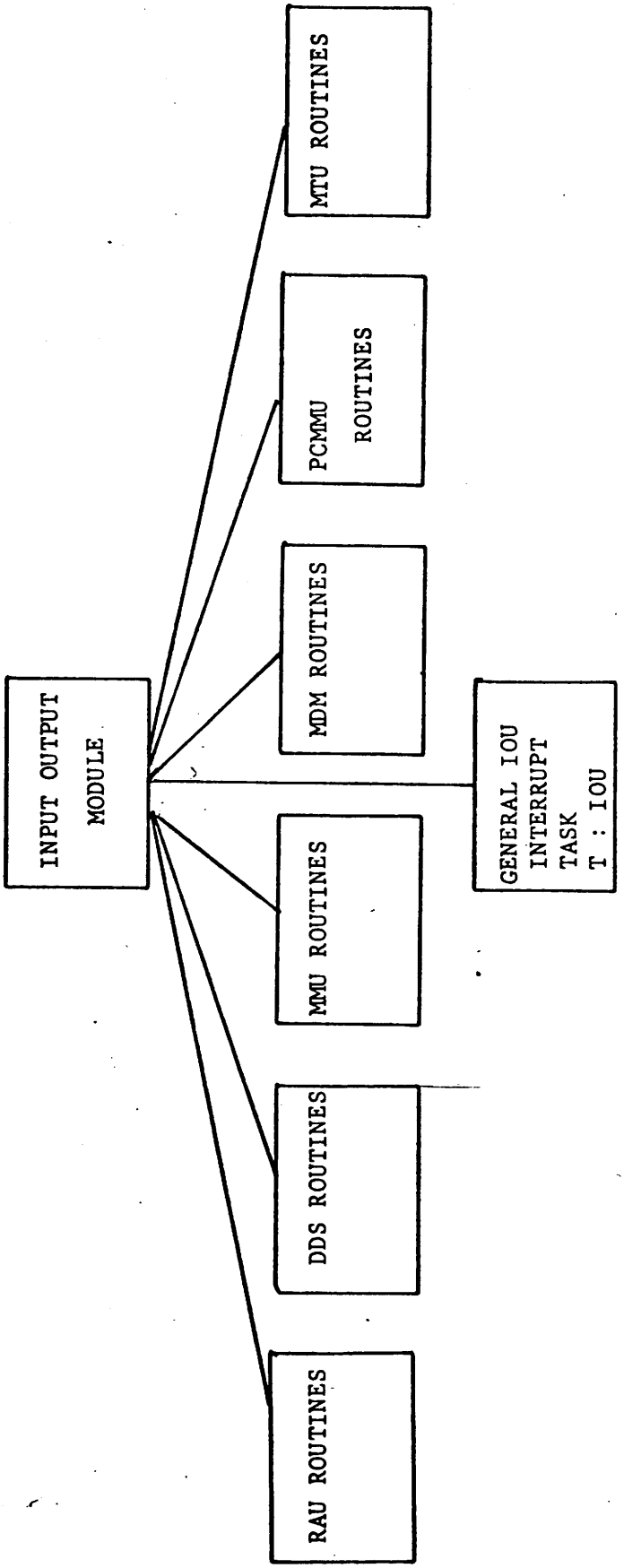


FIGURE 17 : INPUT - OUTPUT MODULE FUNCTION SET DIAGRAM

3.1 - GENERAL IOU INTERRUPT TASK (T:IOU)

All the interrupts generated by the CDMS couplers are gathered by the IOU and trigger only one computer interrupt level called "General IOU Interrupt".

The general IOU interrupt triggers in its turn a S/W task (T:IOU) which is in charge of analysing the IOU IT status word which indicates the coupler which originated this interrupt.

As a result, T:IOU calls the relevant handler H_2 which processes the End of Transfer of the current IO operation.

After completion, the handler returns to T:IOU which then deactivates the interrupt level.

This allows the highest coupler interrupt to trigger again the General IOU interrupt, if such interrupt is pending.

The selection of the highest coupler interrupt is done by the IOU priority chain, according to the following relative priorities :

High = PCMMU
RAU
MDM
MMU
DDS

Low = TIME.

Except for the MDM, all ends of transfer will be processed by the same task T:IOU ; Due to the long time necessary to process 3 inputs and 3 outputs records at the level of T:IOU task, the MDM handler will activate a lower level task to perform the logical processing of MDM records.

3.2 - RAU Routines

3.2.1 - RAU Routine list

T:GML :- This task is in charge of collecting analog and discrete RAU inputs according to predefined IO lists in a system buffer called GMLT, from where the user tasks can fetch data via access routines. T:GML will activate the monitoring tasks according to the sample frequencies.

S:PULS : This routine is in charge of issuing pulses by sending ON & OFF commands.

H:RAU : This routine (handler RAU) is in charge of :

- initiating the transfer on the databus upon request of a task,
- completing the transfer upon detection of the databus coupler end of work interrupt and a waking the requesting task with a report status.

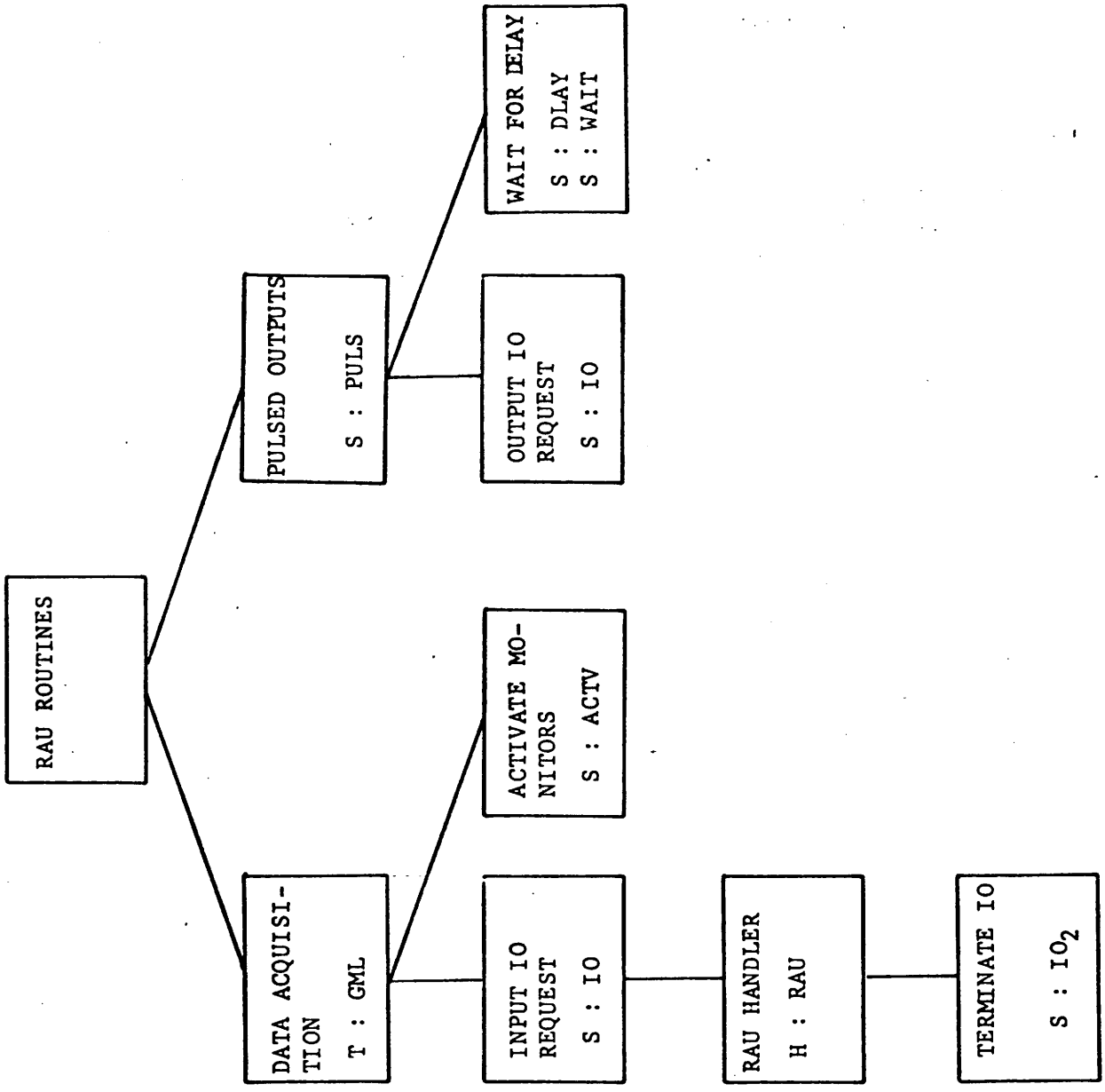


FIGURE 18 : RAU ROUTINES FUNCTION SET DIAGRAM

3.2.2 - Synchroneous acquisition by GML

3.2.2.1 - General description

The SCOS/ECOS operating systems will provide the capability of constantly acquiring the RAU inputs data necessary to subsystem activities.

This capability will be provided by the so-called GML

The data collected by GML will be made available through a common buffer, called GMLT, in particular to the application tasks handling the monitoring functions (referred to as the Monitor tasks).

The design of GML will reflect the requirement for a flexibility in the following two areas :

- GML will support a variable number of sampling frequencies (integer values) with periods which are a multiple of 10 ms

- GML will allow a mix of MP's from several frequencies within the same minor cycle of 10 ms.

This organisation will contribute to accomodation of different mission profiles without changing the code ; the GML processing will be table driven.

These tables will form the output of the offline IO list generator to be provided by ERNO.

3.2.2.2 - GML Processing

- T : GML is a system task connected to an interrupt level allocated at generation time.

- T : GML is activated every 10 ms by the periodic clock task.

FGML is used as a common flag to inform T:GML that it has been activated by a time interrupt.

- T : GML holds an initializing section which is run only once at the first activation of T:GML. This is done to avoid unnecessary processing in the main loop of T:GML every 10 ms.

- The main loop of T:GML will perform the following sequence :

. Reset the FGML flag to indicate that an input is currently performed on the data bus,

. If PCM-MU BSR flag is set, reinitialise the T:GML frequency counters,

. According to the minor cycle, select an IO list,

. Send the IO list via S:IO module to the RAU coupler, thus initiating input for this 10 ms cycle,

. Then deactivate T:GML level.

- T : GML will be reactivated by two possible events :

. The end of IO for data bus input is completed, (indicated by the EVE byte of the IOCB) ; in this case, T:GML will activate the Monitor tasks associated with the frequencies whose counters has reached their maximum value.

Then, if the flag FGML is still reset, T:GML will wait for the 10 ms interrupt before starting its next cycle.

But, if the state of the FGML has been changed in the meantime, it means that the 10 ms interrupt has occured before the normal end of the current cycle (bus/CPU overload) and the next cycle is started without any delay.

. If the end of IO has not occured, it means that T:GML has been reactivated by the 10 ms interrupt before the normal end of the current cycle ; T:GML must not resume its processing before the end of IO on the data bus. The consequence of such a temporary overload will be a slight drift of the time line of the next cycle.

In case of RAU's errors (BITE) the self-test task will be warned.

3.2.2.3 - GML Tables

- T : GML will perform its processing using 3 sets of tables :

1) Frequency descriptor :

A sampling frequency is described by :

. its multiple of 10 ms basic frequency,

. the interrupt level of the Monitor Task to be activated at the completion of one period.

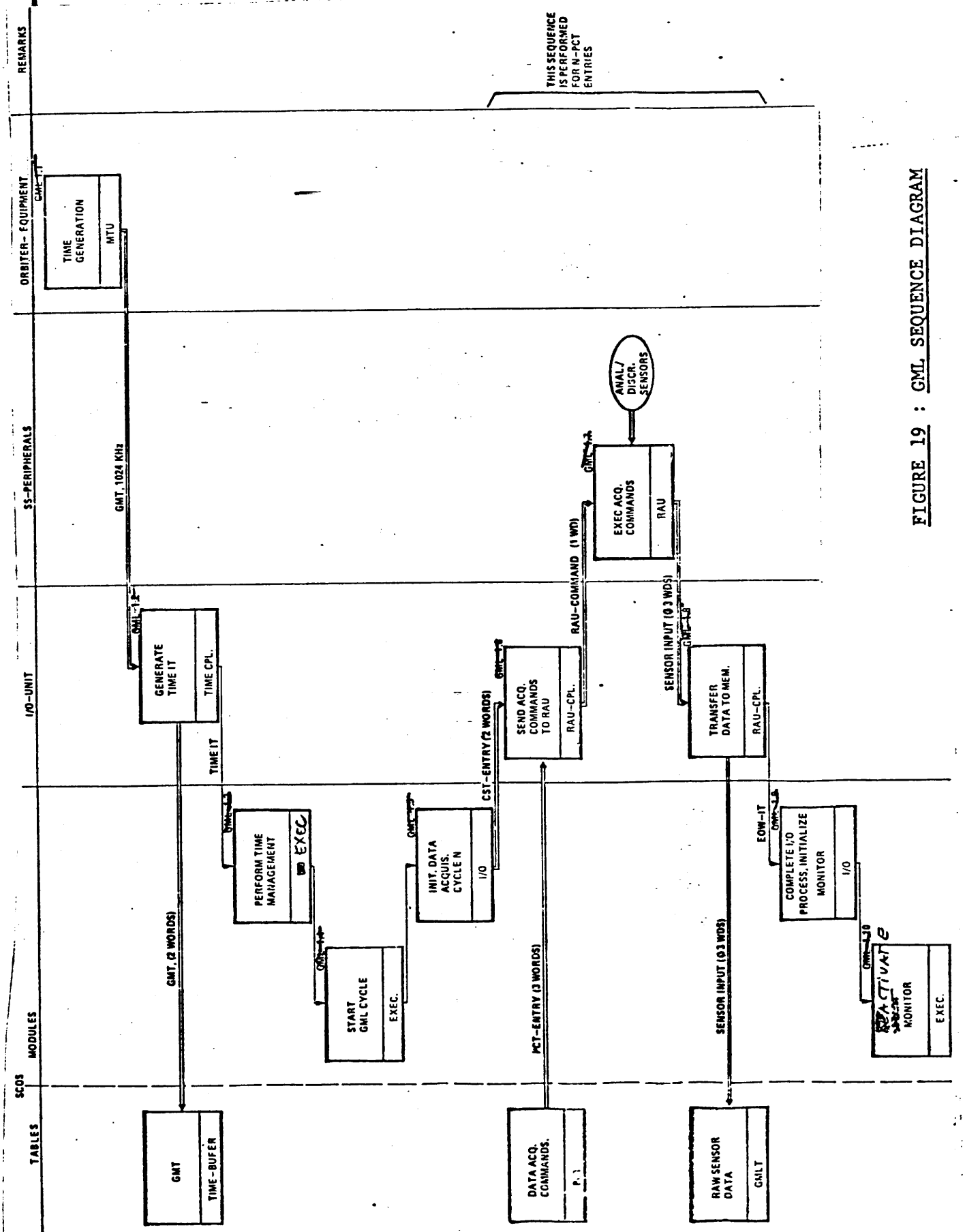
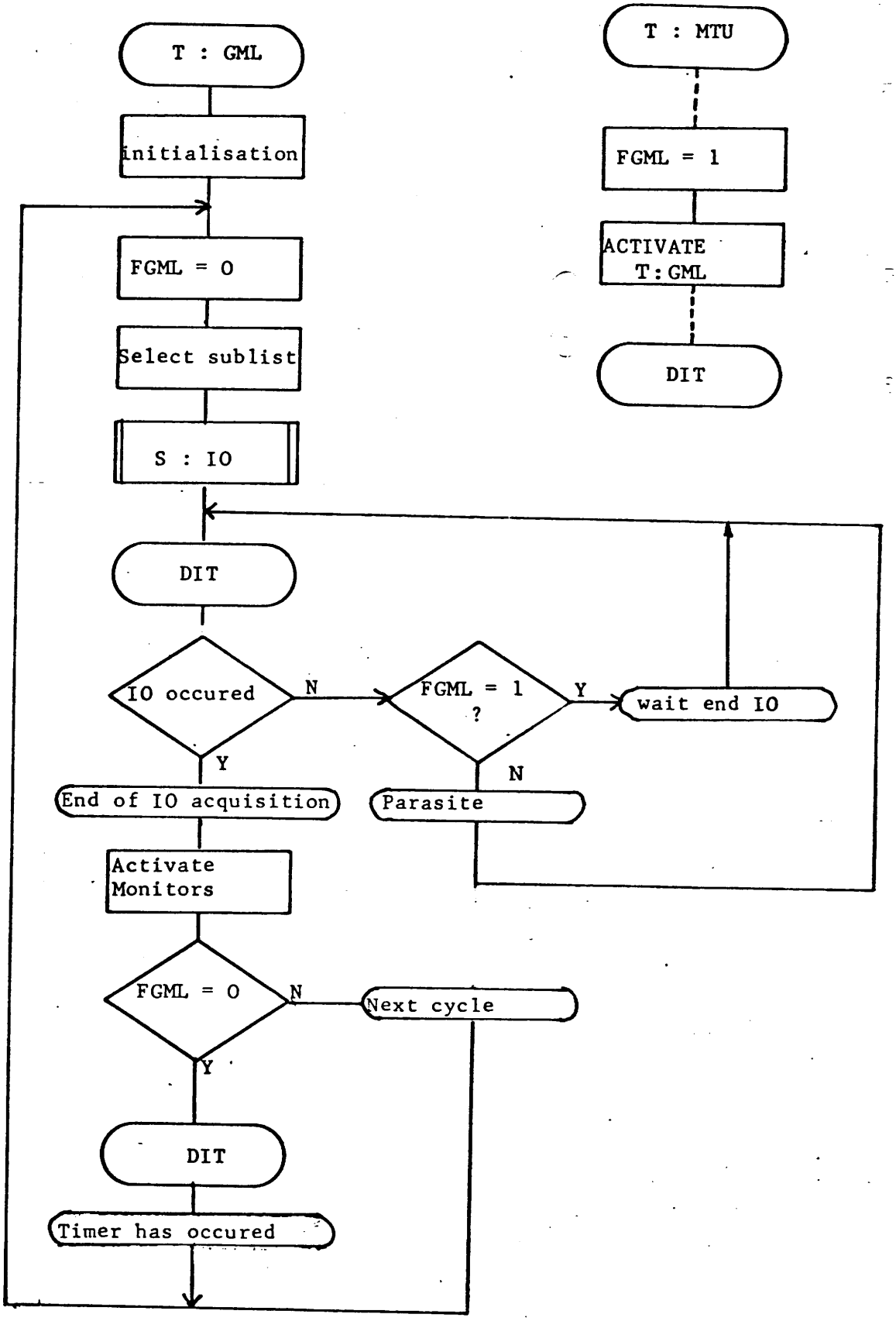


FIGURE 19 : GML SEQUENCE DIAGRAM



GML GENERAL FLOWCHART

2) Sublist descriptor :

. it is the set of the n double words pointing to the n I/O sublists distributed over a complete period (n/100 seconds) ;

These double words are used by T:GML to request an IO transfer for the current minor cycle.

3) I/O sublists :

. the sublists are the triple words generated by the I/O list generator to actually acquire the raw data of one/several MP's. These triple words point to the GMLT buffer. T:GML considers these sublists as constants. Depending on the mission profile, some PCT's may be absent.

. IO sublists are located in the common area.

. the IO list generator will set up the links between the various IO sublists to make up the mix of MP's within a 10 ms time slot. This organisation implies that a frequency is a multiple of the next lower frequency in the ratio = 1, 2, 5, 10.

. See EQ-MA-002 for a detailed H/W interface for the RAU coupler.

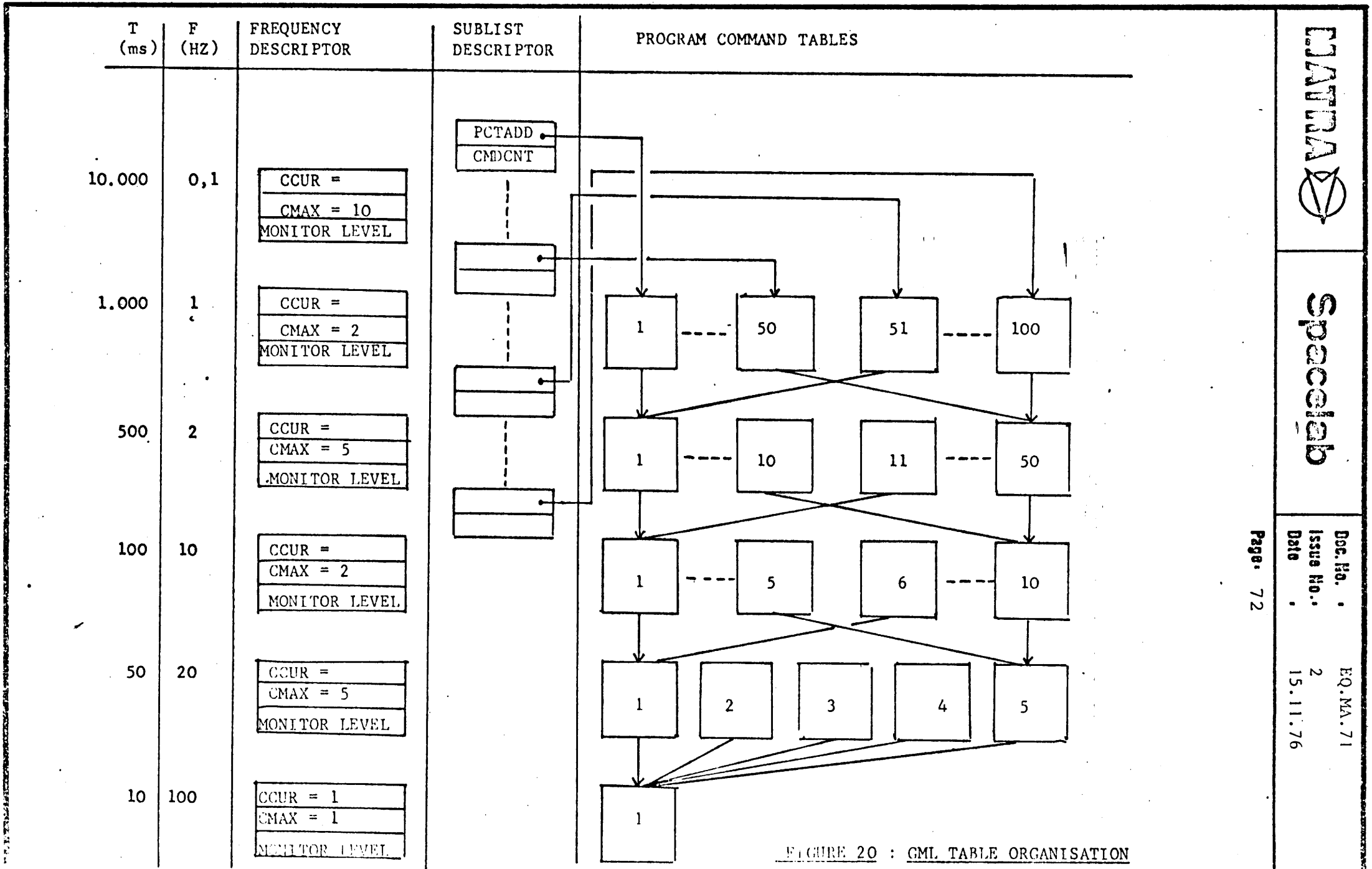


FIGURE 20 : GML TABLE ORGANISATION

3.2.2.4 - Table generation

The tables are generated by the IO list generator.

All PCT's must be generated by the IO list generator as word pointers and will be converted to H/W addresses by T:GML initialization section. This leads to some constraints in GMLT layout (word boundaries).

Every PCT terminated by a double word for PCT linking must have this double word set up.

The addresses in T:LNK table must be generated by the IO list generator as word pointers and will be converted into H/W addresses by T:GML initialization section.

All the tables generated by the IO list generator are of variable length, depending on the number of sampling frequencies available during the mission. The field "TABLE LENGTH" must be set accordingly.

All the tables will be loaded under Z-base and the addresses used by T:GML (to locate these tables) will be copied in the executive tables by the initialization section of T:GML.

The following list describes each table initialization data as output by the IO list generator.

T : CMAX

This table is output by the I/O list generator with the values CMAX corresponding to the sample frequencies used during this mission.

T : CCUR

This table is output by the I/O list generator and CCUR filled with zero.

T : LEV

This table is output by the I/O list generator and filled with the interrupt level number of the associated Monitor task.

The initialization section of T : GML will replace the interrupt level number by the corresponding DVT word to be used as parameter by the WD instruction to activate the Monitor task.

T : LNK

This table is output by the IO list generator with the following fields updated = PCT ADDRESS as word pointer, G bit, L bit, H bit and CMD CNT.

The I/O list generator must also output listings for allowing on-line changes in the IO lists (see § 5.2.6).