THE IP FUNCTION DEVELOPMENT IN THE COMS FLIGHT SOFTWARE

Soo-Yeon Kang, Koon-Ho Yang, Seong-Bong Choi

Korea Aerospace Research Institute
sykang@kari.re.kr

ABSTRACT: The COMS flight software is implemented by ADA language and the high level application program language (APL). The APL is used to implement the Interpreted Program (IP) functions which are running on the processor. The IP functions have been developed for the payload management and monitoring with the automatic on-board operational procedure. The IP allows an easier re-programming if necessary. The ground can load or unload IP separately from ADA code in the RAM flight software. The uploaded IP is interpreted and executed by the Interpreted Program Environment (IPE) which is one of the functions implemented in the RAM flight software. In this paper, we introduce the IP and IPE function.

KEY WORDS: COMS, Flight Software, Interpreted Program, APL, IPE

1. INTRODUCTION

The COMS (Communication Ocean Meteorological Satellite) for the hybrid mission of meteorological observation, ocean monitoring, and telecommunication service has been developed. EADS-ASTRIUM and KARI have worked together for developing the COMS.

The COMS spacecraft bus is based on a generic E3000 Platform, so the COMS flight software running on-board processor also derives from the E3000 product line.

The COMS flight software is implemented by Ada language and the high level application program language (APL). The APL is used to implement the Interpreted Program (IP) functions which are running on the processor.

The IP functions have been developed for the payload management and monitoring with the automatic on-board operational procedures needed to operate the spacecraft. It's a flexible way of customizing the software. An Interpreted Program consists in a sequence of elementary operations autonomously executable on board, so that it can be seen as an on board extension of the ground. The IPs operate independently from the other On-board software functions to which they interface, through the TM/TC processes rather than Ada services or function.

Almost all the functions realised by the IPs are linked to FDIR, autonomously requirements or automatisation of the periodic operational procedures:

- Autonomy and complex timing of execution
- Repetitive actions requiring a precise timing
- Monitoring and FDIR autonomy
- Autonomy for repetitive actions on-station

The IP is compiled by APL compiler and loaded on the flight software running on the processor by TCs. The loaded IP is interpreted and executed by Interpreted Program Environment (IPE) which is a specific software service.

The IPE allows the IPs to run simple programs with as less as possible static or dynamical impacts on the rest of the software, so that the central software safety or ability to fulfil its mission is not jeopardised.

There is no exclusion mechanism between an IP procedure and a central software procedure or between two IP procedures: both can be run simultaneously.

We introduce the IPE which is one of service functions in the RAM flight software as following sections.

2. INTERPRETED PROGRAM ENVIRONMENT

The IPE stores the IPs in the IP storage area, which contains an IP memory image for each loaded IP. The memory is dynamically allocated to load the IP image in the IP storage area when the first telecommand (TC) which includes the loading sequence of the IP is received. If the requested memory can not be allocated, the TC is rejected and the IP loading can not be performed.

Due to 1750 Address Space limitation, the whole physical memory area reserved for IP Memory Image allocation can not be accessed simultaneously by the processor. Physical to logical MMU mapping will be thus dynamically modified before executing each IP, according to the physical pages which have been allocated to at the IP loading time. For this purpose, 4 MMU operand page registers will be dedicated to IP storage area access.

An IP memory is accessed through the IP memory image address table which is indicating the zone of the IP storage area dedicated to each IP. The IP memory image contains the IP management private data, the IP descriptor, the variables, the TM only area, the TM/TC area, the TC only area, the constants and the instructions as shown in the figure 1.

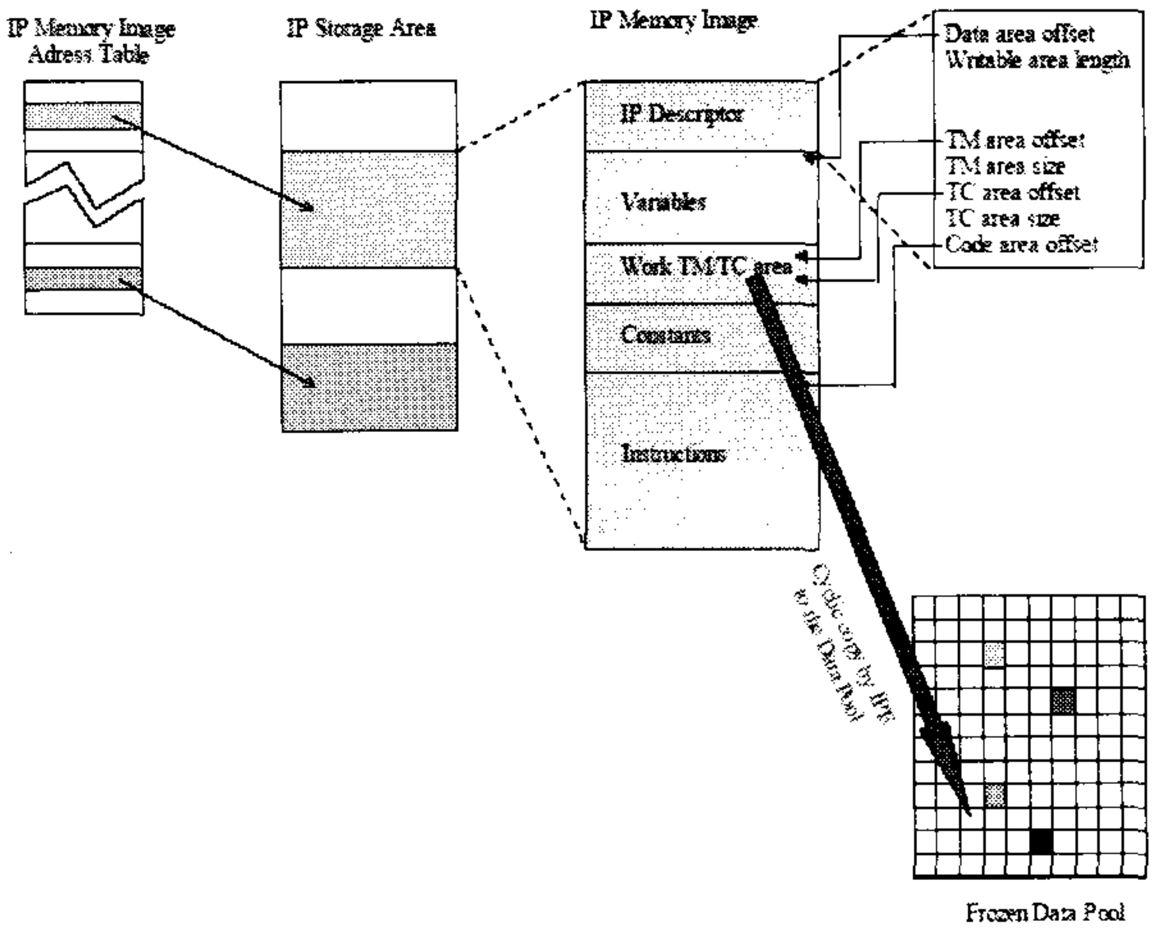

Figure 1.IP Memory Image

One IP descriptor contains all the information related to the loaded IP as follows:

- The ground IP identifier,
- The information of the each IP memory implementation,
- The IP activation parameter by the IP scheduler for the IP execution,
- The IP continuation option in case of SCU reconfiguration,
- The IP entry point which is the offset in the instruction table for the first instruction of the IP main program.

The information of the each IP memory implementation includes the offset of the TM area, offset of the TM/TC area, offset next to the end of the writable area, offset next to the end of the TC area, offset next to the end of the readable area, offset next to the end of the instruction area and the group ID in the data pool of frozen TM/TC area used by the IP. The mentioned offsets are relative to the end of the IP descriptor in the memory image.

Each IP has two private data areas, one is for the variables which are data that can be modified by the IP, and the other is for the constants which are data that can be only read by the IP. The variables and constants are accessed through offsets with respect to a data base address located in the IP descriptor.

The TM area is updated and transmitted into data pool by the IP. The TC area can be modified by the ground in order to provide parameters or commands to the IP. For software and IP restart, the TC area including the TM/TC area is "statically" managed and the TM only area is "dynamically" managed.

These two zones may be overlapped in order to provide an area that can be modified by both ground and IP. The figure 2 shows the TM, TM/TC and TC area.

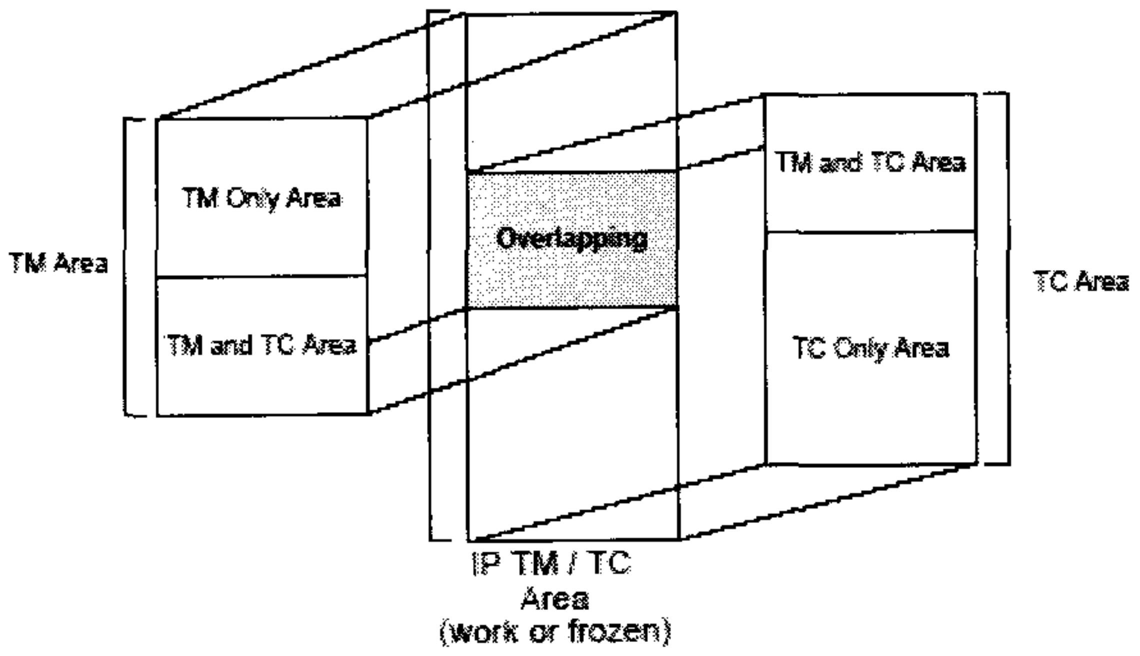

Figure 2.IP TM/TC area

The TM/TC area is a memory area used by the IP to interface with other on-board software applications or commanding by the ground. One work TM/TC area is allocated when the each IP is loaded.

On the request by the IP, some contents of this area zone are copied by the IP into the frozen data pool and are thus available for the TM format.

It means that the IP TM/TC area is duplicated as followings:

- One work TM/TC area which can be modified by the IP at any time.
- One frozen TM/TC area which is part of the data pool is a copy of the work TM/TC area.

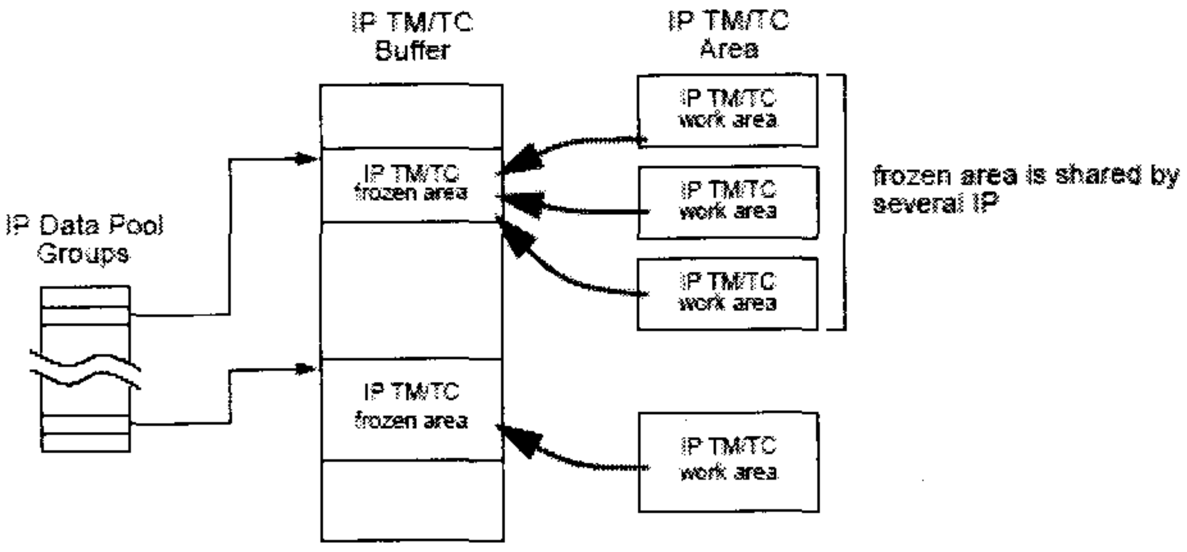

Figure 3.IP TM/TC working and frozen area association

The IP frozen TM/TC area is a group of the data pool. In order to allocate the IP TM/TC buffer in which is part of the data pool, following two methods can be used:

- Reuse an already allocated TM/TC buffer. This may be used by several IPs running exclusively to share the same area in the data pool to reduce memory needs and in the telemetry format to avoid format modification by patch when a newly developed IP is loaded. Only one IP can use it at a given time.
- Dynamic allocation in the free areas of the IP TM/TC Buffer. If the requested memory can not be allocated, the TC is rejected and the IP loading can not be performed.

The copy of the work area in the data pool is performed on the last phase of the VHF cycle with respect to the NF cycle if requested by the IP. This allows an IP to exactly control when its telemetry must be updated. This frozen area is identified in the data pool by a group identifier.

The instruction table is a memory area containing the instructions to be executed by the IP. This can be seen as an array of the instructions. The base address of the instruction table is included in the IP descriptor. All the accesses to the instructions are made through a pointer called the Instruction Pointer which gives the offset of the instruction with respect to the code base address.

IPs can't use the general monitoring services which are provided by the Ada flight software, but have their own monitoring loops implemented. Similarly, there is no automatic management of FIRs between an IP and the Ada software or between two IPs (while this is managed at Ada level). This inter-process FIR compatibility is managed by the IP design.

The IPE can execute IP on the HF, NF and LF tasks or any of their sub-frequencies. Each IP is assigned a fixed time slot for its execution, which is managed by the IPE. This ensures that IP execution cannot randomly affect Ada software load. During an IP execution, the IPE checks the time required for the IP instructions and will only perform those for which it has enough time within the time slot assigned to this IP. If an IP execution step is interrupted because its time slot didn't provide enough time to complete the step, the execution will be resumed on the next cycle.

The IPE can be split in several parts for their functions:

- IP Management: the overall management of the IPs like loading, deleting, starting, stopping,
- IP Scheduling: the activation of the IPs, the frequency of activation handling, and limitation of CPU usage by each IP,
- IP Interpreter: the execution of the IP instructions and of the interface with the rest of the software.

The IPE functions are described in the following sections.

3. IP MANAGEMENT

The Figure 4 shows the IP state transition diagram. All IP gave the same behaviour based on the following automaton.

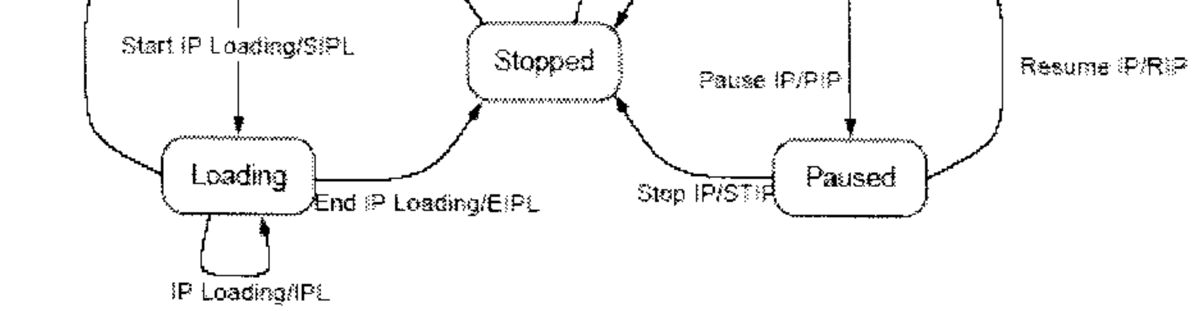

Figure 4.IP state transition diagram

An IP can be in one of the following states:

- Empty: this state indicates that no IP is loaded with the given SW IP identifier. The software IP identifier can be used to load a new IP.
- Loading: this is a transient state during the IP loading. The software IP identifier is reserved, the memory requested by the IP is allocated, but the IP is not yet available for execution until it is fully loaded and the loading is validated by ground.
- Stopped: the IP is not activated by the interpreter.
- Ready: the IP is activated by the interpreter.
- Paused: the IP is paused, an IP can enter this state in order to wait for an external event before resuming its execution, it can be resumed by a TC or by an other IP. In this case it follows it execution from where it has been paused.

The IP management function manages following functions:

- Starting and stopping of IPs,
- Loading IPs from the ground,
- Deleting loaded IPs

4. IP SCHEDULING

The IP can be executed on different activation levels HF, NF and LF. These levels define the priority level associated to the execution of the IP with respect to rest of the SW and the other IPs. The IP scheduler manages the activation of the IP.

The execution of an IP is performed by a cyclic task of the chosen activation level. The execution time allocated to this task must be limited according to its cyclic frequency in order to avoid a cyclic overload, not to endanger the rest of the software. If a whole IP can not be executed on a single cycle, its execution shall spread on several consecutive cycles.

The IP period is used to decrease the CPU load induced by an IP on a given activation level by executing a slice of the IP every N cyclic activation of the underlying task. The IP phase is used to allow the execution of several IP with the same period on different cyclic activation of the underlying task in order to lower the worst CPU load cases. In order to have a deterministic behaviour of the software, the valid IP period values are a subset of all the possible values.

The IP execution on one cycle is limited to a maximum duration. The duration is expressed as a base number of instruction cycle plus an optional delta cycle. The delta is multiplied by a parameter depending of the type of instruction cycle plus an optional delta cycle. The delta is multiplied by a parameter depending of the type of instruction in order to take into account variable duration instructions. These numbers are associated to each instruction and library calls.

Each time an instruction is executed, this number is decreased of the corresponding number of cycle, and when it becomes inferior or equal than zero, the IP execution is suspended or the IP period.

When more than one IP must be executed at the activation of the IP scheduler, they are activated sequential in increasing software IP identifier order.

The global duration of the cyclic task execution the IP is limited by testing the relative time with respect to the beginning of the cycle each time an IP execution is started.

The IP is not started if the relative time is more than a MAX_TIME parameter depending of the activation level. The IP scheduler then tries again to execute the IP on its next activation period.

5. IP INTERPRETER

In this section, we mainly describe the IP instruction format and addressing mode which will be executed by the IP interpreter. The IP interpreter takes charge of the execution of the IP instructions activated by the IP scheduler.

The IP instruction is composed of an instruction word followed by several memory words containing the values of the references of the instruction operands. The IP instruction word is composed of one instruction code indicating the type of processing to be performed and the addressing mode of the operand indicating how the operands are accessed.

The figure 5 shows the IP instruction format used to interpreted by the IP interpreter.

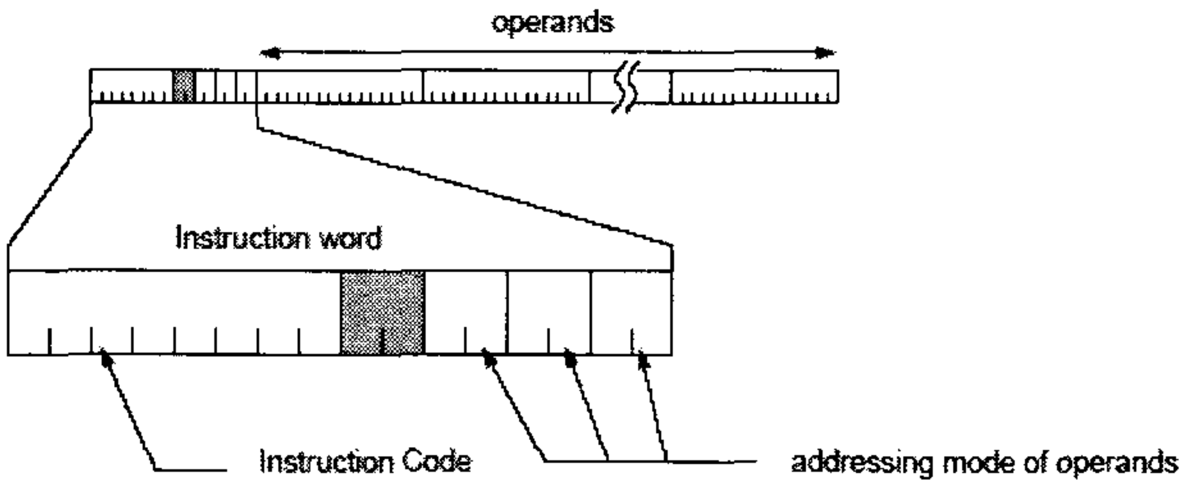

Figure 5.IP Instruction Format

The IP interpreter can handle the following four addressing modes that can be used for the IP instructions: Immediate, Direct, Direct indexed and Indirect indexed.

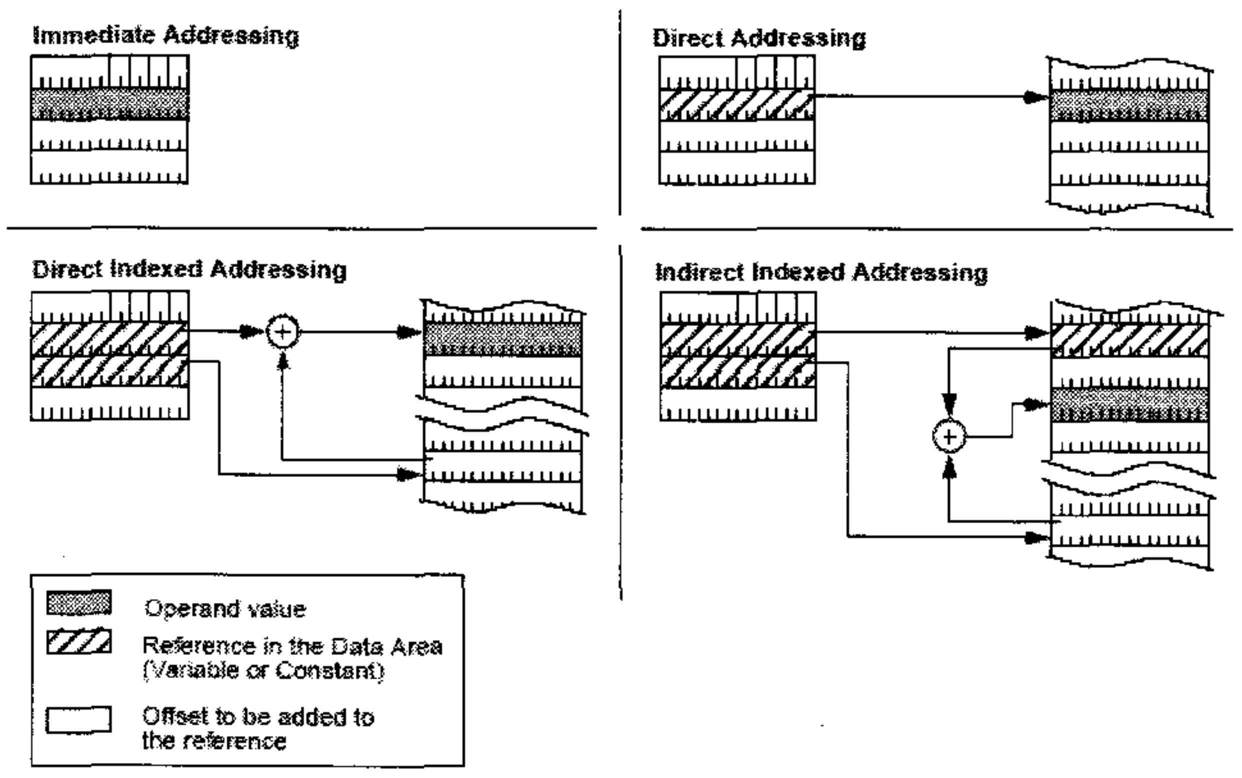
Immediate Addressing
Direct Addressing
Direct Indexed Addressing
Indirect Indexed Addressing
Operand value
Reference in the Data Area (Variable or Constant)
Offset to be added to the reference

Figure 6.IP Addressing Modes

The figure 6 shows the instruction fetch scheme according to the addressing modes.

In case of immediate mode, the operand is fetched in the parameter words following the instruction word.

In case of direct mode, the operand is fetched in the data area, the reference is found in the parameter words following word. This reference is the offset of the operand in data area grouping both variables and constants including TM, TM/TC and TC area.

In case of direct indexed mode, the operand is fetched in the data area, the reference is computed by adding an offset to a base reference. The base reference is found in the parameters and the offset is found in the data area at a reference given in the parameters.

In case of indirect indexed mode, the operand is fetched in the data area, the reference is computed by adding an offset to a base reference. Both base reference and offset is found in the data area at a reference given the parameters.

In case of both direct indexed and indirect indexed addressing mode, the resulting offset is checked against the authorized data areas of the IP.

6. CONCLUSION

In this paper, we introduce the IP function written in the application program language (APL) and IPE which manages, interpreters and executes the IPs. The IP provides the flexible way of customizing the software to operate independently from the central flight software.

The IPE function is provided to manage, execute and interpreter the IPs as one of service functions running on the processor.

7. ACKNOWLEDGEMENTS

The COMS project is sponsored by Korea Government with the following institutions involved: MOST (Ministry of Science and Technology), KMA (Korea Meteorological Administration), MIC (Ministry of Information and Communication), MOMAF (Ministry of Maritime Affairs and Fisheries).