ILLIAC III COMPUTER SYSTEM MANUAL:
INTERRUPT UNIT

Volume 1: Logical Design

by

Lakshmi N. Goyal
Robert M. Lansford

March 5, 1970

MASTER

RECEIVED BY DTIC APR 27 1970

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Report No. 386

ILLIAC III COMPUTER SYSTEM MANUAL:
INTERRUPT UNIT*

Volume 1: Logical Design

by

Lakshmi N. Goyal
Robert M. Lansford

March 5, 1970

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

*Supported by Contract AT(11-1)-1018 with the U.S. Atomic Energy Commission.

Table of Contents

1. INTRODUCTION
 - 1.1 Initial Guidelines
2. TYPES OF COMMUNICATION
 - 2.1 Processor - IU - Processor
 - 2.2 Processor - IU
 - 2.3 TP - IØP
 - 2.4 IØP - TP
 - 2.5 TP - TP
 - 2.6 Processor - IU
3. LOGICAL IMPLEMENTATION
 - 3.1 The IU Complex
 - 3.2 Unit Commands
 - 3.2.1 Mnemonic Field
 - 3.2.2 Message Field
 - 3.2.3 Functional Description of Unit Command Categories
 - 3.2.3.1 Processor to Processor Message (PROPPROM)
 - 3.2.3.2 Processor to IU Response (PROIUR)
 - 3.2.3.3 Processor-Acknowledgement (PROACK)
 - 3.2.3.4 Processor to IU Message (PROIUM)
 - 3.3 IU-Request Cycle
 - 3.3.1 Transmission of the Unit Command to the IU via the INBUS
 - 3.3.2 Transmission of Reply (Status) to Requesting Processor via the OUTBUS

3.3.3 Execution of Unit Commands

3.3.3.1 PROPROM

3.3.3.2 PROIUR

3.3.3.3 PROACK

3.3.3.4 PROIUM

3.3.4 Parity Error and IU Operation

3.3.4.1 Recognition of Parity Error by the IU

3.3.4.2 Recognition of Parity Error by a Processor

4. PROCESSOR-IU MESSAGE INSTRUCTIONS

4.1 Diagnostic Instructions

4.1.1 Load Registers/Flip-flops

4.1.2 Read Registers/Flip-flops

4.1.3 Miscellaneous Diagnostic Instructions

4.1.3.1 Set Maintenance Halt

4.1.3.2 Reset Maintenance Halt

4.1.3.3 Run

4.1.3.4 Position

4.1.3.5 Execute

4.2 Non-diagnostic Instructions

4.2.1 Read Clock

Appendix A

Appendix B

Bibliography

1. INTRODUCTION

The Illiac III Computer¹ is a multiprocessor computer system and consists of processors and units, interconnected by an exchange net.² Figure 1.1 shows a schematic diagram of the Illiac III Computer. However, as is evident from the figure, no direct path is provided for communication from one processor to another. It is rather the job of the Interrupt Unit (IU) to provide for orderly transmission of messages (interrupt commands or response data) from one processor to another. The IU is essentially a 'store and forward' message processor and synchronizes the asynchronously originating processor to processor requests. It may, however, be noted that the IU does not process the system interrupts but is only a message router. System interrupts are processed entirely in the processors themselves.

Figure 1.1 - Schematic of Illiac III Computer System

1.1 Initial Guidelines

In investigating the various forms which the IU might take, an attempt was made to adhere to a few basic guidelines.

1.1.1 The IU should be as fast as is reasonable. It should not tie up the Exchange Net for any appreciable length of time (long relative to Exchange Net access time).

1.1.2 As a necessary result of 1.1.1, the IU should be as simple as possible. To this end, functions which already exist in various processors should be utilized if at all possible.

1.1.3 IU should need an absolute minimum of information on the status of any given processor in order to carry out its functions.

1.1.4 A corollary to 1.1.3 is that each processor should control its own destiny to the greatest extent possible rather than depend upon the IU.

2. TYPES OF COMMUNICATION

There are two broad categories of types of communication between a processor and the IU. They are:

- a. Processor → IU → Processor
- b. Processor → IU

2.1 Processor-IU-Processor

This communication is necessary to provide a path for transmission of interrupt commands and/or response data from one processor to another through the IU. Without great difficulty one can visualize the following processor-to-processor communication.

- 2.1.1 TP → IOP so that a TP can exert control over the I/O operations.
- 2.1.2 IOP → TP which allows the IOP to notify a TP of certain conditions arising in the IOP, e.g. a channel program has ended.
- 2.1.3 TP → TP This is necessary for multiprocessor/multiprogramming capabilities of the system. This allows one TP to order another TP to switch over to a new job of possibly higher priority.

2.2 Processor-IU

This communication is necessary in cases where a message may be explicitly directed to the IU for execution entirely within the IU itself. In particular the interrupt unit contains a real time clock which can be read by any processor.

The above mentioned communication paths will now be examined in detail below; first to spell out clearly the desired action and then to extract the IU requirements.

2.3 TP - IOP

This category consists of directives which the TP may issue to an IOP. These include:

SIO	Start I/Ø
HIO	Halt I/Ø
TIO	Test I/Ø

Each of these messages is one word long. (See IOP Manual).

From the TP's point of view, these instructions are used to send a message to an IOP. Because of the specificity of the I/Ø devices involved, this type of message must be directed to a specific IOP. It should also be noted that each processor proceeds independently of the others, so that the IOP to which the message is directed may not be able to accept it immediately.

For these instructions, it is necessary for the TP to receive a return status word (Acknowledgment word) before proceeding to the next instruction. The purpose of the status word is to let the TP know the disposition of the message.

From this description, we may draw a number of conclusions:

2.3.1 Following the issuance of a message, the TP should go into a wait state. In this state the TP will only recognize a reply from the IOP.

Arguments for this are as follows:

2.3.1.1 The TP needs the return of the information before it can complete the instruction.

2.3.1.2 It is then unnecessary to generate an interrupt on return.

2.3.1.3 Waiting insures that the TP will be able to accept the reply immediately from the IOP via the IU.

2.3.1.4 This automatically resolves several priority problems which otherwise arise.

2.3.2 Because the IOP may not be able to immediately accept the message, the IU should provide message buffering.

2.3.3 As the result of 2.3.2, the IU must have the capability to alert the (specific) IØP for which a message awaits. Furthermore it is conceivable that several messages might await the same IØP.

2.3.3.1 A means must exist to select one of possibly several messages to forward to the IØP.

2.3.3.2 The alerting condition ($IØR_k$, $k = 0, 1$) seen by the IØP should be the logical sum of the conditions ($IØR_{jk}$, $j = 0, 1, 2, 3$; $k = 0, 1$) at the individual TP message buffers (TP_{jBR}).

2.4 IØP - TP

The purpose of this message is to generate, at a TP, what is commonly known as an I/Ø interrupt. It may be issued by an IØP to signify that a channel program has terminated or that a spurious condition has arisen, etc. This message is also one word long (See IØP Manual).

This message type is not to be confused with the reply generated by the IØP during TP - IØP communication (Acknowledgment) and none is received in reply by the IØP from the TP.

Although this message is issued by a specific IØP, it is in general of no importance which TP receives the message. This is so because due to multi-programming/multi-processing, the task which issued the I/Ø request is very likely not running at the time the channel program terminates. Furthermore, a given TP may now be masked for the interrupt condition and thus unable to respond.

In addition to the problems already discussed in connection with TP - IØP communication, there is one peculiar to the IØP. It is possible that several conditions could arise in the IØP for which it would like to transmit a message to a TP before the IU can forward the first of these to a TP. Some buffering must be provided, but it would be advantageous if it were in the IØP.

To this we conclude the following:

2.4.1 The IU must provide a buffer for IØP messages. It, in fact, seems advantageous to associate a buffer with each of the possible originating (requesting) IØP processors.

2.4.2 An IØP Buffer Register Full (BFk, k = 0, 1) should be provided for each IØP. Once a message has been transmitted to the IU, the BFk line may be raised to prevent further transmission to the buffer. By using this feature, messages to be sent to the IU may be queued within the IØP. Thus, most buffering is done within the IØP - which allows the IØP to completely manage its outgoing messages.

2.4.3 No destination processor will be specified; instead an external interrupt signal, in this case called IØI (I/Ø Interrupt), will be transmitted to all TP's.

2.4.3.1 Only those TP's will respond which are enabled (unmasked) for the condition. Thus the IU need not know the (dynamic) status of each TP.

2.4.3.2 Which TP will obtain access to the IU will be determined by the Exchange Net. If the IØI signal is then removed, the remaining TP's will have only been delayed for one EN access.

2.4.3.3 The alternatives to broadcasting, in general, take longer and require much more information by the IU.

2.4.4 The IØI line to the TP's should be the logical sum of the two IØIk (k = 0,1) lines as both IØP's may have messages for the TP's.

2.4.4.1 No acknowledgment is returned to the IØP. It may then go about its business as soon as the message is received by the IU.

2.5 TP - TP

This message is issued by one TP to another TP. Since Illiac III is a multi-processor configuration with multiprogramming capabilities, it becomes necessary as a normal operation to stop the processing of one task and to start another. Each processor (TP) must be capable of performing this operation for any other processor.

The activation of a TP is initiated by the execution of an 'Activate TP' instruction by another TP. The 'Activate TP' instruction causes the TP executing the instruction to make a TP Interrupt request to the Interrupt Unit. This request specifies which TP is to be interrupted. Thus corollaries of this facility are:

2.5.1 The Interrupt Unit should have the facility for informing the appropriate TP about this interrupt command, by providing external interrupt signals called TPjAct ($j = 0,1,2,3$).

2.5.2 Following the request to the IU, the requesting TP will wait, as in a TP - IOP case, for the acknowledgment word which specifies the disposition of the message transmitted. Upon arrival of the acknowledgment word, the requesting TP which initiated the 'Activate TP' instruction enters the Final Control sequence and then proceeds to the next instruction. To the TP receiving the activate command from the IU, the command is a type of external interrupt. The exact operations which are performed by this TP depend on the state of the receiving TP. If the TP is operating on a task, the command is treated as a normal external interrupt. If the TP is idle, a normal external interrupt is performed except that no interrupt data is stored. In either case, the interrupt handler is responsible for detecting the type of interrupt and transferring control to the Supervisor Reactivation Procedure.

For more details, see Sections 5.6.3.3 and 5.6.6.5 of the TP Manual, Vol. 3.

2.6 Processor - IU

Such a request includes non-diagnostic instructions like 'Read the real time clock' or diagnostic instructions like 'Read IU buffer register' etc. A detailed discussion of these requests is deferred to Section 4.

3. LOGICAL IMPLEMENTATION

In this section a brief description of the necessary register and bus arrangements is given. This is followed by the definition of the Unit Commands necessary for communicating between processors and the IU. Finally, the concept of an IU-request cycle is given and the four Unit Commands are discussed in terms of the IU-request cycle.

3.1 The IU Complex

In Fig. 3.1.1, the internal buffer registers, message paths and external interrupt signals are shown. Not surprisingly, all of the features discussed in Section 2 appear. There are six buffer registers - one for each of the processors, and a clock counter-register for the real time clock. Each register is four bytes long and holds the message (which cannot exceed 4 bytes in length) from the corresponding processor. Associated with each IOP internal buffer register are two flip-flops IØIO, BFO and IØI1, BF1 respectively which provide external interrupt signals to the IOP's. Similarly for each of the TP internal buffer register TP1BR($i = 0,1,2,3$), there are three TPijACT ($j = 0,1,2,3, j \neq i$) flip-flops and two IØRik ($k = 0,1$) flip-flops which generate TPjACT (TP Activate) external interrupt signals to TP's and IØRk ($k = 0,1$) (I/Ø Request) external interrupt signals to the IOP's.

Internal Processor Buffers are loaded directly from the INBUS and are directly transferred to the OUTBUS, after an appropriate choice of the register.

Fig. 3.1.2 illustrates the previously discussed lines (external to the Exchange Net) which provide external interrupt signals from the IU to the processors.

2/26/70

Section 3.1 - 2/3

Fig. 3.1.1 - Schematic Diagram of the Illiac III Interrupt Unit

Figure 3.1.2 - External Interrupt Signal Lines from IU to Processors

3.2 Unit Commands

Since the processors interface with the exchange net for transmission of messages, it is necessary to define the Unit Commands for the IU. These Unit Commands inform the Interrupt Unit about the type of messages being sent by the processor so that the IU can take the desired action.

Since the unit command is carried by the INBUS word (hereinafter referred as INWORD) of the exchange net interface, the format of the unit command has to be consistent with the INWORD format. Control byte and data field bytes of the INWORD carry respectively the mnemonic field and message of the Unit Command. The format is shown in figure 3.2.

Figure 3.2 - Unit Command Format

Table 3.2 gives the coding for the Unit Commands and Processor Addresses.

<u>Command Type</u>	<u>Command Code</u>	<u>Address Field (Command Variant)</u>			
		Source		Destination	

PRO-PRO Message (PROFROM)

TP → IØP	1 0 0	0/1	0/1	X	0/1
IØP → TP	0 1 0	X	0/1	X	X
TP → TP	1 1 0	0/1	0/1	0/1	0/1

PRO-IU Response (PROIUR)

TP: IØI	0 1 1	X	X	X	X
IØP: IØRO, IØR1	1 0 1	X	0/1	X	X
TP: TPjACT (j=0,1,2,3)	1 1 1	0/1	0/1	X	X

PRO-IU Message (PROIUM)*

TP, IØP → IU	0 0 1	0/1	0/1	0/1	0/1
--------------	-------	-----	-----	-----	-----

PRO-Acknowledgment (PROACK)

TP, IØP → TP	0 0 0	X	X	0/1	0/1
--------------	-------	---	---	-----	-----

Note: X implies a 'don't care'.

* See section 3.3.3.4 for interpretation of this field

Table 3.2 - IU Coding for the Unit Commands
and Processor Addresses

3.2.1 Mnemonic Field

The mnemonic field of the Unit Command consists of a Command Code (3 bits) followed by processor address bits (4 bits). It has a format shown in Figure 3.2.1.

Figure 3.2.1 IU Unit Command Mnemonic Field

The Unit Commands can be grouped under four broad categories as shown below.

3.2.1.1 Processor to Processor Message (PROPPROM)

3.2.1.1.1 TP → IOP (I/O request)

3.2.1.1.2 IOP → TP (I/O interrupt)

3.2.1.1.3 TP → TP (TP activate)

3.2.1.2 Processor to IU Response (PROIUR)

3.2.1.2.1 TP response to IOI (I/O interrupt)

3.2.1.2.2 IOP response to IOR (I/O request)

3.2.1.2.3 TP response to TPACT (TP activate)

3.2.1.3 Processor Acknowledgment (PROACK)

3.2.1.4 Processor to IU Message (PROIUM)

Function of each of these categories is briefly mentioned in section 3.2.3 and each command is later on discussed in detail in section 3.3.

3.2.2 Message Field

Message field carries information for either the destination processor or the Interrupt Unit. In the case of PROPROM and PROACK, the message field consists respectively of interrupt command and processor acknowledgment word for the processor designated in destination address bits of the mnemonic field. In the case of PROIUM, it carries the instructions to be executed in the IU itself.

3.2.3 Functional Description of Unit Command Categories

3.2.3.1 Processor to Processor Message (PROPPROM)

These unit commands tell the IU that the accompanying message is an interrupt command for onward transmission to another processor. The following types of communication are provided:

TP	→	IØP	(I/Ø request)
IØP	→	TP	(I/Ø interrupt)
TP	→	TP	(TP activate)

The IU stores the message in the appropriate buffer (corresponding to the requesting processor), and after decoding the unit command informs the appropriate destination processor(s) by raising the external interrupt signals.

3.2.3.2 Processor to IU Response (PROIUR)

When the requested processor is in a position to handle the external interrupt initiated by the IU, this unit command is used by the requested processor to request the message waiting for it in the IU. This is used by a TP to get the message in response to IØI (I/Ø interrupt), or TPjACT (j = 0,1,2,3) (TP activate) and by an IØP to get the message in response to IØRk (k = 0,1) (I/Ø request).

3.2.3.3 Processor-Acknowledgment (PROACK)

After the TP's and the IØP's have received the message from the IU in response to a TPjACT (j = 0,1,2,3) or IØRk (k = 0,1), the requested processor needs to send an acknowledgment to the requesting processor. This acknowledgment word informs the waiting requesting processor about the disposition of the message (interrupt command) sent by it to the requested processor.

3.2.3.4 Processor to IU Message (PROIUM)

This unit command category tells the IU that the accompanying message is an instruction which would be executed locally in the IU itself. These commands are for reading the real time clock or any other diagnostic commands issued by the TP to the IU.

The detailed action taken by the IU on the receipt of each of these unit commands from the processor are discussed in terms of an IU unit cycle in the next section.

3.3 IU-Request Cycle

An IU-request cycle consists of three phases executed in the following order:

- 1) A processor (TP, IOP) transmits the unit command and the message, if any, to the IU.
- 2) IU decodes the Unit Command and executes it.
- 3) IU transmits a reply to the requesting processor.

Phase 2 is unique for each Unit Command, but Phase 1 and 3 are common to all the four Unit Commands. In the discussion below, Phases 1 and 3 are discussed first, followed by a discussion of Phase 2.

3.3.1 Phase 1: Transmission of the Unit Command to the IU via the INBUS

A processor requests the Exchange Net (EN) for access to the IU. It is the responsibility of the EN to resolve all conflicts arising from attempted multiple access (from different processors) of a common unit.

Once access is granted by the EN, the Unit Command and the message if any, are sent in the control byte and data field of the EN INBUS word to the IU. The IU stores the Unit Command and the message in the appropriate buffer register. Following this, the processor may break the connection for access to the EN. However, the processor waits for a 'status of transmission' indication from the IU in Phase 3 of IU request cycle.

In order to protect itself against premature release of the EN, the IU will immediately declare itself busy with respect to the local exchange. This will prevent a second processor from obtaining access before the IU can handle the message.

3.3.2 Phase 3: Transmission of Reply (Status) to Requesting Processor via the OUTBUS

The message to be transmitted is first assembled. The IU then requests access to the EN OUTBUS and once access is obtained the message is transmitted to the requesting processor. The message mainly consists of status information, in the OUTBUS control byte, about any parity error during transmission, malfunction of the IU or any illegal Unit Command, etc. However, in the case of the PROIUM (Processor to IU message) Unit Command, results obtained by executing the instruction specified in the data field of INBUS word in Phase 1 are sent in the data field of OUTBUS word, along with the status. In case of PROIUR, the contents of a proper buffer register are sent in the data field of the OUTBUS word, along with the status in the control byte.

Following transmission, the IU may release the path and declare itself free to receive a new Unit Command and message in control byte.

3.3.3 Phase 2: Execution of Unit Commands

Following sections describe in detail the action taken by the IU during the execution phase of each of the Unit Commands.

3.3.3.1 PROPPROM (Processor to Processor Message)

The action essentially consists of determining the address of the destination processor and then setting the appropriate external interrupt signals. The various types of communication involved are:

TP → IØP

IØP → TP

TP → TP

3.3.3.1.1 TP → IØP (I/Ø Request Command)

The interrupt command here is an I/Ø request destined for a particular IØP. The IU advises the designated processor about the message waiting for it in a TP buffer register by raising the IØR0 or IØR1 external interrupt signals. This is done by setting IØRjk (where $j = 0,1,2,3$; $k = 0,1$) signals corresponding to the j th Source TP and k th destination IØP. The IØRk ($k = 0,1$) is the logical sum of the conditions IØRjk at the individual TP buffer registers. This allows for the possibility that several messages can wait one I/Ø processor. This is necessary because various processors act independent of each other. Thus, clearing one buffer and its associated IØRjk will not necessarily remove the IØRk signal to the IØPk.

3.3.3.1.2 IØP → TP (I/Ø Interrupt Command)

The message is an I/Ø interrupt command issued by an IØP to a TP. Since neither the IØP nor the IU can select an appropriate TP because of lack of knowledge of the dynamic status of the TP masks and priorities, the IØI (I/Ø interrupt) external interrupt signal is broadcast to

all TP's. Depending on the Source IØP, IØIO or IØI1 is raised. However, IØI - the signal broadcast to all the TP's - is the logical sum of both IØIO and IØI1. In addition, the IU raises the corresponding BFO or BF1 (Buffer Full) condition for the storage buffer, to indicate to the requesting IØP that it should not request IU again unless the previous request has been processed (indicated by the dropped BFk line). It is thus the responsibility of the IØP to check the activity condition of BFk line each time it wishes to transmit a message to IU.

3.3.3.1.3 TP → TP (TP Activate Command)

The source TP buffer register contains a TP ACTIVATE interrupt command. Just like TP → IØP, this is meant for a specific TP and the designated TP is informed of the interrupt by raising TPjACT (j = 0,1,2,3) external interrupt signal. This is done by setting TPijACT (i, j = 0,1,2,3; i ≠ j) signal corresponding to the ith source TP and the jth destination TP. TPjACT (j = 0,1,2,3) is the logical sum of the TPijACT (i = 0,1,2,3; i ≠ j) conditions at the individual TP buffer registers, TPiBR.

It may be noted here that TP ACTIVATE command is fully specified by the Unit Command in Control byte and there is nothing useful in the data field of the INBUS word. So the source TP buffer contains mainly garbage. However, making it look like a TP → IØP command (as if data field had valid information) is simply for simplicity and uniformity of control.

At this point, each of the above enters the sequence described in Section 3.3.2. This completes one IU cycle.

3.3.3.2 PROIUR (Processor to IU Response)

Each of the Unit Commands and messages transmitted to the IU under PROPROM results in the IU raising some external interrupt signal to the designated processor.

Because of the asynchronous nature of the processors, the destination processor may not be in a position to respond to the message (interrupt command) at the time the message is passed to the IU. For this reason, all such messages are first stored in the processor buffer registers in the IU and destination processors are then informed via the external interrupt lines. Now the designated processor can respond at its convenience and request the message pending for it in the IU. This request for the pending message from the IU is the function of this Unit Command. The responses will be considered in the order in which the messages were issued under PROPROM in Section 3.3.3.1.

3.3.3.2.1 IOP Response to an IOR (I/O request) Signal

Provided that higher priority tasks are not pending, the IOP_k ($k = 0,1$) will transmit the Unit Command in the control byte of INBUS word (data field will have no useful information) to the IU (as in Section 3.3.1). This command is stored in the Unit Command Register and decoded.

In executing this command, the IU control finds out which of the TP buffer registers contains the message for the requesting IOP_k by checking the IOR_{jk} ($j = 0,1,2,3; k = 0,1$) associated with the TP_jBR. Due to independent operation of processors, more than one message (I/O request) may be pending for the same IOP. Hence some priority algorithm must be used to service the request. The IU checks the buffer registers in the order TPOBR, TP1BR, TP2BR, TP3BR. The first register with the correct IOR_{jk} ON is selected for transmission to the OUTBUS. (In case of multiple

messages pending, the clearing of one IØRjk will not remove the IØRk interrupt signal seen by the IØPk. Once this I/Ø request is taken care of by the IØP, another I/Ø request from another TP can be handled by sending a PROIUR type Command afresh).

Once the access to the Exchange Net is obtained, the contents of the selected TPjBR are returned to the IØPk as described in Section 3.3.2, and the corresponding IØRjk is reset.

3.3.3.2.2 TP Response to an IØI Interrupt Signal

Conceptually, the TP's response is the same as above. However, the logical action is complicated by the fact that the signal is broadcast to all TP's. TP's able to respond to the IØI will attempt to access the IU. The TP allowed to do so by the Exchange Net will then transmit this unit command in order to obtain the message (I/Ø interrupt command). The IU scans the IØIO and IØII signals associated with IØPOBR and IØPlBR respectively. Since normally the TP does not have any preference for an I/Ø interrupt command from IØPO or IØPl, a priority algorithm is used to service the request: IØPO is given higher priority over IØPl.

Once the IØPOBR or IØPlBR is selected for transmission to the requesting TP, access is requested to the exchange net. When the access is granted, the contents of IØPkBR are transmitted and corresponding IØIk and BFK interrupt conditions are reset.

NOTE: In response to broadcast conditions, the responding processor (a TP) should only request a path to the IU as long as the IØI condition remains valid. Once IØI line is dropped, the TP's should withdraw their request for access to the IU. This will release those TP's which may have simultaneously responded to the IØI interrupt signal.

3.3.3.2.3 TP Response to a TPjACT Interrupt Signal

This is exactly identical to the response of an IOP to an IOR interrupt signal discussed in the previous section except that TPijACT will have to be checked when the requesting processor is TPj.

It should be remarked here that although TPiBR does not contain any information required by TPj ($i \neq j$) to handle this interrupt command (section 3.3.3.1.3), the TPj should go through the process of requesting the message for TPj. This is the only way for the IU to know that TPj has handled the TPjACT command so that TPijACT can be reset. This makes the execution control for this Unit Command identical to that of other unit commands of the PROIUR type Unit Command.

3.3.3.3 PROACK (Processor Acknowledgment)

The final IU cycle for a processor to processor communication is an acknowledgment sent from the destination processor to the (waiting) requesting processor (TP) in the case of

TP → IØP

TP → TP .

No acknowledgment (from TP to an IØP) is transmitted in the case of an IØP → TP Communication.

After transmitting the message in the PROPROM Unit Command, the TP goes into a wait state and waits for an acknowledgment word from the destination processor (IØP or TP). Clearly the acknowledgment message must be directed to a specific TP. In the execution of PROIUR commands, two source address bits from the unit command field (in PROPROM) are sent along with the interrupt command message from the source TP to the responding processor (IØP, TP). The responding processor utilizes these two bits in its Unit Command destination processor address field while sending the acknowledgment to the Source processor through the IU.

The acknowledging processor will access the exchange net in the normal fashion, if need be. If the acknowledging processor can reply quickly enough, it need not relinquish the exchange net path obtained under PROIUR.

Since the source processor (TP) waits for the acknowledgment from the destination processor, the source processor would not send any other message for transmission to another processor during the period of its wait. Hence the source processor buffer register in IU can be used to store the acknowledgment word from the acknowledging processor. This way, the IØP need not monitor the BFK line before sending the acknowledgment because the message does not go through the IØP buffer registers in IU. This allows for the situation that the IØP can still handle the message from another processor although its earlier interrupt command to a TP (IØP → TP) has not yet been processed.

Upon the receipt of acknowledgment Unit Command from the acknowledging processor, the message field (acknowledgment word in data field of exchange net INWORD) of the Unit Command is placed in the TP buffer register identified in the destination address field of Unit Command.

Since the processor to which acknowledgment is to be sent is waiting, there is no need to send an external interrupt signal to that processor. Instead, it is directly transmitted to the waiting processor by accessing the EN OUTBUS.

3.3.3.4 PROIUM (Processor to IU message)

This category of unit commands consists of instructions which are executed totally within the IU. The IU contains a real time clock. The instructions are of two types, namely, diagnostic and non-diagnostic. Diagnostic instructions are issued by the maintenance processor via a TP. The non-diagnostic instruction is for reading the real time clock and is issued by a TP or an IØP.

The usual interpretation of address field in the control byte of INWORD has no significance for this category of unit commands. Instead, these bits are used to interpret the types and formats of instructions of PROIUM unit command category.

In terms of the IU-request cycle, phase 1 is identical to that discussed in section 3.3.1 except that for non-Load Register type diagnostic commands, the message in the data field of the INWORD is stored in the 8-bit Instruction Register (IR). In phase 2, the command in the IR is decoded and executed and then finally the control enters phase 3 of the IU-request cycle.

For detailed interpretation and description of various instructions in this category, see section 4.

3.3.4 Parity Error and IU Operation

The IU checks parity whenever it receives an INWORD from the exchange net. Also it generates parity when it sends any message on the OUTWORD to the processor. Thus two types of parity errors can occur:

- i) IU recognizes a parity error while receiving a message from a processor.
- ii) Processor recognizes a parity error while receiving a message from the IU.

The action taken by either the IU or the processor, as the case may be, on the recognition of a parity error (PE) is discussed in the following pages.

3.3.4.1 Recognition of Parity Error by the IU

The control byte of the INWORD does not have a parity bit, but each of the four data bytes in the data field of the INWORD have a parity check bit associated with it. Hence whenever the unit command indicates useful valid information will be found in the data bytes of the INWORD, parity error must be taken into consideration. Action taken by the IU on recognition of a parity error, for each type of the unit command is as follows:

PROPROM:

There are two possible cases here. One, when the Source processor is other than an IOP and the second when the Source processor is an IOP.

Source Processor $\neq$ IOP

IU informs the Source processor (TP) of a parity error through the OUTWORD control byte (bit 5 of OUTBUS) and does not inform the destination processor that a message is waiting for it in the IU buffer register, i.e. the external interrupt flip-flop will not be set. This way, the source processor can transmit the message again.

Source Processor = IØP

Since IØP is not in a position to retransmit the message (due to lack of time), this message (an I/Ø interrupt) should be transmitted to the destination processor (a TP). The message data is flagged as being invalid due to parity error. It is up to the TP to decide what to do with the invalid data, whether to ignore the interrupt command or not. IØP, however, is informed of a parity error having occurred during the reception (by IU) of the message from IØP.

PROACK:

The acknowledgment word properly flagged as being invalid is transmitted to the proper destination. The acknowledging processor is informed of parity error.

PROIUM:

The requesting processor is informed about the parity error and the instruction is not executed.

PROIUR:

Since the unit command to the IU does not have any useful information in the data bytes of INWORD, the parity error is ignored.

3.3.4.2 Recognition of Parity Error by a Processor

Although this section has nothing to do with the IU operation, it is given here for the completeness of discussion.

PROFROM:

No IU to processor message transmission takes place through the exchange net and hence the question of parity error does not arise.

PROACK:

The receiving processor may ignore the parity error or may like to issue the commands again, i.e. to initiate the whole process again.

PROIUM:

Same as in PROACK.

PROIUR:

This is the most difficult of all to resolve. A processor does not tell back the unit whether there was any parity error during the reception of message (or data) from the IU. Since after the transmission of message (from another processor) in response to PROIUR command, IU destroys that message, this message is irretrievably lost and the message originating processor would never know about it being lost except by noting that the originating processor will never get the acknowledgment possibly because the destination processor would ignore the data. But there is nothing else that can be done because processors don't respond to units about status of data received from a unit.

In the case of other units, there is no problem because the processor can ask for the computation being done again. But in case of IU, this message comes from another processor whose identification is not known to the present processor.

4. PROCESSOR-IU MESSAGE INSTRUCTIONS

This section defines the various diagnostic and non-diagnostic instructions executed locally in the IU. The format, coding and function of each such instruction is described in detail.

4.1 Diagnostic Instructions

These instructions are issued by the maintenance processor via a TP and are used for debugging the IU. These instructions are a relevant subset of the diagnostic commands defined for the TP (Section 4.8 of TP manual). For ease of discussion, the instructions can be thought of as being divided into two subgroups, according to their format:

- a) Load/Read Registers
- b) Miscellaneous .

For subgroup a), the instruction code is defined in the control byte of INWORD and for subgroup b) the code is given in the data bytes of the INWORD. Each of these subgroups are further discussed in detail in the following sections.

4.1.1 Load Registers/Flip-flops

These load instructions cause the designated register to be loaded from the data lines or flip-flops to be set. The IU contains six buffer registers and a set of 24 external interrupt signal flip-flops and a few control flip-flops. The desired register is indicated in the three bits (bit 7,8,9) of the control byte of the exchange net INWORD. The data to be loaded is placed in the data bytes of the exchange net INWORD. For setting the flip-flops, a '1' is placed in the bit position reserved for that particular flip-flop(s). This way, more than one flip-flop can be set at any time and thus provide a flexible diagnostic command. The format for these instructions is given in figure 4.1.1.

Figure 4.1.1 - 'Load Register' Diagnostic Instruction Format
Table A-2 in appendix A shows the code assignment for each load registers/flip-flops instruction.

(In terms of a IU-request cycle after the unit command is received, the instruction is decoded and the data is directly loaded in the appropriate buffer register. At this point, the sequence enters the phase 3 for 'status of information' transmission described in section 3.3.2.)

4.1.2 Read Registers/Flip-flops

These instructions cause the designated register or all the signal flip-flops (control and external interrupt signal flip-flops) to be read out onto the data lines in data field of OUTBUS word. In case of 'Read Flip-flops', instead of reading a particular flip-flop (F/F), all the F/F's are read out and the TP can choose to look at any particular one by masking in the data received. The format is given in figure 4.1.2. Table A-3 in appendix A shows the code assignment for the various registers.

Figure 4.1.2 - 'Read Registers' Diagnostic Instruction Format

4.1.3 Miscellaneous Diagnostic Instructions

This set includes all those instructions, the code for which is given in the first of the data bytes of the INWORD. The various instructions are:

Set Maintenance Halt
Reset Maintenance Halt
Run
Position
Execute

A certain combination (0111) of the bits 6, 7, 8, 9 of the control byte is used to distinguish this subgroup of instructions from other instructions in the PROIUM category. The format for these diagnostic instructions is given in figure 4.1.3.

Figure 4.1.3 - 'Miscellaneous' Diagnostic Instruction Format

The code for each instruction is given in Table A-4 in the appendix A.

(In terms of IU-request cycle, phase 1 is identical to that described in section 3.3.1 except that the Instruction register IR is loaded from the first of the data bytes in the INWORD. In phase 3, as mentioned earlier, the result, if any, along with the status information, is sent in the OUTWORD over the OUTBUS. Phase 2, though simple, is unique to each instruction.)

Each instruction is now described in detail.

4.1.3.1 Set Maintenance Halt (SMH)

The SMH instruction causes the Maintenance Halt flip-flop, MHLT, to be set to one. This automatically activates the maintenance stop signals to every control point which utilizes them.

4.1.3.2 Reset Maintenance Halt (RMH)

The RMH Command causes the Maintenance Halt flip-flop, MHLT, to be reset to zero. This automatically turns off all maintenance stop signals to every control point which utilizes them.

4.1.3.3 Run

This instruction is used along with SMH instruction for single stepping through the control points. For more details, see section 4.7.2.3 of TP manual, and reference 3.

4.1.3.4 Position

The POSITION instruction causes the IU to send to the maintenance processor via the TP, its currently active control point position within the control logic. Since there are not very many control points in the IU control, each control point is uniquely identified by one position in the data field of OUTBUS word.

More details will be given in reference 3.

4.1.3.5 Execute (EXEC)

The EXEC instruction executes the unit command specified by the bits 11-17 in the first data byte of the INWORD. Execution mainly consists in setting or resetting of appropriate external signal flip-flop. It is to be noted that no data transfer from the IU to the processor will take place, as a result of execution of the Unit Command.

4.2 Non-diagnostic Instructions

These instructions are issued by a processor (TP or IOP) to the IU for execution locally in the IU. Only one such instruction has been defined for reading the real time clock contained in the IU. This instruction is 'Read Clock' and is defined in section 4.2.1.

4.2.1 Read Clock (RDCLK)

This instruction is used to read the 24 hour accounting clock. This instruction can be issued by a TP or an IOP.

The contents of the clock at the time of execution of this instruction is read right justified into the data bytes of the OUTBUS Word.

It is desirable that this instruction be distinct from the Read Registers diagnostic command. This instruction does not need any operand. So the instruction code is placed in the four bits (bits 6-9) of the control byte. Its format and code is shown in figure 4.2.1.

Figure 4.2.1 - Format and Code for 'Read Clock'

APPENDIX A

Appendix A gives the coding for the various Unit Commands and instructions executed locally in the IU. It also shows the formats for the Unit Commands, and diagnostic and non-diagnostic instructions executed locally in the IU.

Figure A.1 - Unit Command Format

<u>Command Type</u>	<u>Command Code</u>	<u>Address Field (Command Variant)</u>			
		Source		Destination	

PRO-PRO Message (PROPRM)

TP → IØP	1 0 0	0/1	0/1	X	0/1
IØP → TP	0 1 0	X	0/1	X	X
TP → TP	1 1 0	0/1	0/1	0/1	0/1

PRO-IU Response (PROIUR)

TP: IØI	0 1 1	X	X	X	X
IØP: IØRO, IØR1	1 0 1	X	0/1	X	X
TP: TPjACT (j=0,1,2,3)	1 1 1	0/1	0/1	X	X

PRO-IU Message (PROIUM)*

TP, IØP → IU	0 0 1	0/1	0/1	0/1	0/1*
--------------	-------	-----	-----	-----	------

PRO-Acknowledgment (PROACK)

TP, IØP → TP	0 0 0	X	X	0/1	0/1
--------------	-------	---	---	-----	-----

Note: X implies a 'don't care'.

* See section 3.3.3.4 for interpretation of this field

Table A-1 - IU Coding for the Unit Commands
and Processor Addresses

Figure A.2 - 'Load Register' Diagnostic Instruction Format

<u>Code</u> (bits 3-9 of EN INWORD control byte)	<u>Instruction</u>
0 0 1 0 0 0 1	Load IØPOBR
0 0 1 0 0 1 0	Load IØP1BR
0 0 1 0 0 1 1	Load TPOBR
0 0 1 0 1 0 0	Load TP1BR
0 0 1 0 1 0 1	Load TP2BR
0 0 1 0 1 1 0	Load TP3BR
0 0 1 0 0 0 0	Set flip-flops

Table A-2 -- Code Assignment for 'LOAD Registers'
Diagnostic Instructions

2/26/70

Section A - 6/9

Figure A-3 - 'Read Register/Flip-flops' Diagnostic Instruction Format

<u>Code</u> (bits 3-9 of EN INWORD control byte)	<u>Instruction</u>
0 0 1 1 0 0 1	Read IØPOBR
0 0 1 1 0 1 0	Read IØP1BR
0 0 1 1 0 1 1	Read TPOBR
0 0 1 1 1 0 0	Read TP1BR
0 0 1 1 1 0 1	Read TP2BR
0 0 1 1 1 1 0	Read TP3BR
0 0 1 1 0 0 0	Read flip-flops

Table A-3 — Code Assignment for 'READ Registers'
Diagnostic Instructions

2/26/70

Figure A-4 - 'Miscellaneous' Diagnostic Instruction Format

<u>Code</u>								<u>Instruction</u>
(bits 10-17 of exchange net INWORD)								
0	0	0	0	1	0	0	0	Set Maintenance Halt
0	0	0	0	1	0	0	1	Reset Maintenance Halt
0	0	0	0	1	0	1	0	Run
0	0	0	0	1	0	1	1	Position
1	0/1	0/1	0/1	0/1	0/1	0/1	0/1	Execute

Table A-4 -- Code Assignment for 'Miscellaneous'
Diagnostic Instructions

APPENDIX B

Summary of Exchange Net and Processor Requirements

In this appendix are summarized those requirements which the design just specified imposes on the various processors and the EN. This is not to imply that those requirements are different from those at present -- some are, some are not. Rather it is to serve as a check list for the processor designers to use in evaluating the proposed IU and to emphasize those requirements that must be considered if the design is to be implemented as part of Illiac III. For implementation details the reader should consult 3).

B.1 Exchange Net

B.1.1 Bit 9 of the exchange net INBUS is presently used only in the case of the AU's. It must be extended from the processors to the IU.

B.1.2 The IU must have a means of loading its outbus Processor Identification Register (PIR) so that it may reroute messages.

B.1.3 The exchange net must function in such a way that if a request is made, that request may be withdrawn, provided that the exchange net has not yet provided the path, i.e.: if multiple processors are requesting the same unit.

B.2 I/O Processor (IØP)

B.2.1 Two lines BFK and IØRk ($k = 0,1$) are provided, external to the exchange net, to IØPk to serve as the Buffer Full and I/O Request lines, respectively.

B.2.2 The IØP must check for the corresponding BF line to be low ($BF = 0$) before it attempts to transmit an I/O interrupt to the TP.

B.3 Taxicrinic Processors (TP)

B.3.1 Two lines are provided from the IU to each TP. These lines are the I/O Interrupt (IOI) and TP Activate (TPjACT) lines respectively.

B.3.2 The TP can be masked for either or both of the above conditions. Thus the TP must only react if the condition is unmasked (enabled).

B.3.3 Should the TP 'see' the conditions simultaneously, the standard priority scheme assigned to the interrupt sequence of the TP will resolve the problem. (Although the IU cannot set the conditions simultaneously, the TP may not be able to 'look' until such a time as they appear simultaneous.)

B.3.4 When the TP is waiting following a TP - TP or TP - IOP message, the TP must not react to either of the above conditions.

B.3.5 When the TP is waiting, it should have some means of determining that an acknowledgment has not been received within a reasonable length of time and hence that a malfunction or missing processor must be indicated.

B.3.6 If the TP has responded to an IOI condition and the condition is dropped before the TP can obtain access to the IU (another TP serviced the request), the TP should remove its request.

BIBLIOGRAPHY

1. McCormick, B. H. and Lansford, R. M., "The Illiac III Programming Manual", Dept. of Computer Science, University of Illinois, March 12, 1968.
2. Krabbe, Paul S., "A Discussion of Illiac III Processor-Unit Communication via the Exchange Net", File No. 790, Dept. of Computer Science, University of Illinois, March 4, 1969.
3. Goyal, L. N., "The Hardware Realization of the Interrupt Unit of Illiac III", Dept. of Computer Science, University of Illinois (in progress).

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.

Report No. 386
COO-1018-1199

2. TITLE

LOGICAL DESIGN OF THE INTERRUPT UNIT OF ILLIAC III

3. TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:
 Title of conference _____
 Date of conference _____
 Exact location of conference _____
 Sponsoring organization _____
☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

Mr. Lakshmi N. Goyal, Research Assistant
Dr. Robert M. Lansford, Research Associate

Organization

Department of Computer Science
University of Illinois

Signature

Lakshmi N. Goyal

Date

March 5, 1970

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.