Table of Contents

Notices

z/Architecture is a registered trademark of International Business Machines Corporation.

Introduction

This manual describes a set of macros and their usage providing program synchronization of with externally events without formal interrupt service routines.

The following macros, found in the maclib macro library directory, belong to the sync category. The Input/Output Control Block (IOCB) is a common structure used by many of the sync macros.

Macro	Requires IOCB	Description
CPUWAIT	no	The program waits for an externally triggered event.
ENADEV	yes	Ready a device for use by the program.
IOCB	not applicable	Assembled device specific input/output control block.
IOCBDS	not applicable	Maps the IOCB fields.
IOINIT	no	Prepare the system for input/output operations
IOTRFR	yes	Calculates the data transferred by an I/O operation.
RAWAIT	yes	Waits for an externally triggered event from a device independent of input/output operations.
RAWIO	yes	Initiates and waits for completion of input/output operations.

This dummy section is created by the IOCBDS macro.

DSECT	Description
IOCB	Maps the IOCB fields.

How to Use This Document

If this is the first time you are reading this document, it is recommended that you read the “HOWTO...” sections of interest first. This gives the background helpful in understanding where each macro fits into event synchronization as implemented by the Stand Alone Tool Kit (SATK).

There is an assumption that the reader has knowledge of basic mainframe input/output operations involving channel program execution under control of a legacy operating system.

Specifics relating to bare-metal processing for channel programs is not assumed. See the “HOWTO Perform Input/Output Operations” section when using the sync category of macros

for this purpose. Refer to actual assembled macro expansions for the actual programming techniques used.

For information related to architecture levels, refer to either SATK.odt or SATK.pdf. Architecture levels apply to nearly all macros.

CPUWAIT

Source file: macros/CPUWAIT.mac

Macro Format:

```
[label]   CPUWAIT [EXT=[YES|NEW|label]
                  [,EPSW=label]
                  [,ESA=[YES|NO|label]]
                  [,IO=[YES|NEW|label]
                  [,IPSW=label]
                  [,ISA=[YES|NO]]
                  [,AM=[24|31|64]]
                  [,CHAN=[254|n]]
```

The CPUWAIT macro allows the program to wait, in-place, for either an external interruption or input/output interruption or either without the use of a formal interrupt service routine.

Assembly Considerations:

- Preceding ARCHLVL and ARCHIND macros are required.
- A base register must have been established before using CPUWAIT.

Execution Considerations:

- During the period in which program instruction execution ceases, interruptions are enabled for the requested interruption types. EXT= has not been omitted or IO= has not been omitted.
- Following the event, external and input/output interruptions are disabled by the PSW introduced by the occurrence of the event.
- Recognition of an event always causes the CPU to continue execution within the CPUWAIT macro. This allows the macro to perform any cleanup actions before it returns control to the program via a branching operation or introduction of a new PSW as identified by EPSW= or IPSW=.

Label Field Usage:

The label field, if present, defines an assembly label associated with the first instruction generated by the macro.

Positional Parameters: None

Keyword Parameters:

Keyword	Default	Description
AM	24	The address mode following the event when either EXT=YES or IO=YES.
CHAN	254	When waiting for an I/O event in assembly architecture levels 1 or 2, identifies the channels from which an I/O interruption is excepted. Ignored by other assembly architecture levels.
EPSW	None	If EXT=YES, the PSW identified by the label is introduced as the new PSW. Otherwise, EPSW is ignored. Ignored if EXT omitted.
ESA	YES	Controls how the external new PSW assigned storage location is managed. When EXT is present, a PSW will be placed at the assigned storage location of the External Interrupt New PSW. • YES causes the content of the assigned storage location to be preserved by the macro before the CPUWAIT external PSW is placed at the assigned storage location, and upon recognition of an event, the saved content is restored. • NO causes the content of the assigned storage location to be replaced by the CPUWAIT External New PSW and remains following the occurrence of either event. • label – causes the content of the assigned storage location to be saved at this location and restored from it rather than one supplied by the CPUWAIT macro when ESA=YES. Ignored if EXT is omitted.
EXT	None	Specifies how control is returned to the program in the event an external interruption occurs. • YES – control passes to the program following the CPUWAIT macro with the condition code set to 1. • NEW – control passes to the program as dictated by the PSW identified by the EPSW= parameter. • label – control passes to the program location. If omitted, external interruptions remain pending.
IO	None	Specifies how control is returned to the program in the event an input/output interruption occurs. • YES – control passes to the program following the CPUWAIT macro with the condition code set to 2. • NEW – control passes to the program as dictated by the PSW identified by the IPSW= parameter. • label – control passes to the program location. If omitted, input output interruptions remain pending.
IPSW	None	If IO=YES, the PSW identified by the label is introduced as the new PSW. Otherwise, IPSW is ignored. Ignored entirely if IO is omitted.
ISA	YES	Controls how the input/outputl new PSW assigned storage location is managed. When IO is present, a PSW will be placed at the assigned storage location of the Input/Output Interrupt New PSW. • YES causes the content of the assigned storage location to be preserved by the macro before the CPUWAIT input/output PSW is placed at the assigned storage

Keyword	Default	Description
		location, and upon recognition of an event, the saved content is restored. • NO causes the content of the assigned storage location to be replaced by the CPUWAIT External New PSW and remains following the occurrence of either event. • label – causes the content of the assigned storage location to be saved at this location and restored from it rather than one supplied by the CPUWAIT macro when ISA=YES. Ignored if EXT is omitted.

Programming Note:

CPUWAIT represents the minimum required to interact with external or input/output interruptions.

If only input/output operations are being performed, it is recommended that RAWIO and RAWAIT be used instead of CPUWAIT. Both RAWIO and RAWAIT internally use CPUWAIT, but both provide additional housekeeping and error handling functions specific to input/output devices.

When a formal interrupt service routine is not being used (which is why CPUWAIT is being used), it is assumed a trap PSW has been placed in the external and input/output new PSW assigned storage locations. Use of ESA=YES, ESA=label, ISA=YES, or ISA=label will allow the trap PSW to be restored following the recognized event.

ENADEV

Source file: macros/ENADEV.mac

Macro Format:

```
[label]   ENADEV ready,fail
                 [,REG=n]
                 [,CLS=n]
                 [,CSNS=[YES|NO]]
```

The ENADEV macro prepares a device for I/O operations. The device is identified by means of the device's assembled IOCB structure.

Assembly Considerations:

- Preceding ARCHLVL and ARCHIND macros are required.
- The IOCB DSECT must be defined preceding the ENADEV macro.
- Addressing of the device's IOCB structure must be established through the IOCB DSECT preceding the ENADEV macro.
- A local base register must have been established preceding ENADEV.

Execution Considerations:

The IOCB associated with the device must be completely initialized preceding use of ENADEV.

Three registers are always required:

- a local base register (other than 0 or 1)
- a register addressing the device's IOCB structure (other than 0 or 1)
- general register 1 is altered by ENADEV.

For architecture levels 5-9, the following additional register considerations apply:

- A register other than 0 or 1 is required for addressing the channel subsystem specific structures (the REG= keyword parameter).
- Another optional register is needed if the interruption subclass of the device is being set (the CLS= keyword parameter).

Label Field Usage:

The label field, if present, defines an assembly label associated with the first instruction generated by the macro.

Positional Parameters:

1. The ready positional parameter identifies the label to which control is passed if the

device is ready for use.

2. The `fail` positional parameter identifies the label to which control is passed if the device fails to be made ready.

Keyword Parameters:

Keyword parameters are only used when the assembly architecture level is 5-9, using the channel subsystem. All other architecture levels ignore these keywords if present.

Keyword	Default	Description
CLS	None	A register containing the subschannel interruption subclass of the device. The subclass is a value between 0 and 7, inclusive, and must be in bits 26-28 of a 32-bit registers or bits 58-60 of a 64-bit register. If omitted the subchannel interruption subclass will be 0.
CSNS	NO	Whether use of concurrent sense is to be enabled for the device. • YES – enables concurrent sense • NO – leaves concurrent sense disabled. Once concurrent sense is enabled, it remains enabled.
REG	required	The register available for addressing channel subsystem structures.

IOCB

Source file: macros/IOCB.mac

Macro Format:

```
[label]   IOCB  dev,
                [,CERR=[X'3F'|n]]
                [,UERR=[X'D3'|n]]
                [,WAIT=[X'80'|n]]
                [,CCW=[0|label]]
                [,KEY=[0|n]]
                [,SCHIB=[0|label]]
                [,IRB=[0|label]]
                [,ORB=[0|label]]
                [,FLAG=flags]
```

The Input/Output Control Block (IOCB) structure, used by most of the macros described in this manual, is assembled by the IOCB macro. Each device for which input/output operations by the program uses sync category macros must have its own dedicated IOCB. The IOCB maintains status information related to ongoing actions taken by the program using the device.

The IOCB is mapped by the IOCBDS macro within the IOCB DSECT.

The details embedded in the structure may be assembled or established during program execution, or some mixture, provided the IOCB is completed prior to use by the ENADEV macro.

Assembly Considerations:

- Preceding ARCHLVL and ARCHIND macros are required.

Execution Considerations:

- The details embedded in the IOCB may be assembled or established during program execution, or some mixture, provided the IOCB is completed prior to use by the ENADEV macro. Refer to the “IOCB Structure” section for the relationships between IOCB macro parameters and the IOCB DSECT fields.

Label Field Usage:

The label field, if present, defines an assembly label associated with the start of the assembled IOCB.

Positional Parameters:

1. The dev positional parameter is required. It provides the user identification of the device, depending upon architecture level:

- Levels 1-4 – the device's channel/unit address as a hexadecimal self-defining term in the CCUU format, or
- Levels 5-9 – the device's device number as a hexadecimal self-defining term.

Keyword Parameters:

Parameters used by all architecture levels

Keyword	Default	Description
CERR	X'3F'	Specifies the solicited channel status conditions, as a hexadecimal self-defining term bit mask, recognized as errors for the device. Mask bits in **bold** font are enabled by the default mask. • X'80' – Program-controlled Interruption • X'40' – Incorrect Length • **X'20' – Program Check** • **X'10' – Protection Check** • **X'08' – Channel-data Check** • **X'04' – Channel-control Check** • **X'02' – Interface-control Check** • **X'01' – Chaining Check**
UERR	X'D3'	Specifies the solicited unit status conditions, as a hexadecimal self-defining term bit mask, recognized as errors for the device. Mask bits in **bold** font are enabled by the default mask. • **X'80' – Attention** • **X'40' – Status modifier** • X'20' – Control-unit end • **X'10' – Busy** • X'08' – Channel end • X'04' – Device end • **X'02' – Unit check** • **X'01' – Unit exception**
WAIT	X'80'	Specifies the unsolicited unit status conditions, as a hexadecimal self-defining term bit mask, recognized as satisfying waiting upon the device. Mask bits in **bold** font are enabled by the default mask. • **X'80' – Attention** • X'40' – Status modifier • X'20' – Control-unit end • X'10' – Busy • X'08' – Channel end • X'04' – Device end • X'02' – Unit check • X'01' – Unit exception

Parameters Used by Parallel Channel Operation (Levels 1-4)

Keyword	Default	Description
CCW	0	Specifies the address of the first channel command word for the next executed RAWIO

Keyword	Default	Description
		macro. Contributes to the Channel-Address Word.
KEY	0	Storage access key used by the channel for next usage of RAWIO macro. Contributes to the Channel-Address Word

Parameters Used by Channel Subsystem Operation (Levels 5-9)

Keyword	Default	Description
CCW	0	Specifies the address of the first channel command word in the embedded ORB for the next RAWIO macro. Ignored if the ORB is not provided by the IOCB macro.
FLAG	none	Specifies the embedded ORB's flags. If omitted, and the ORB is embedded in the IOCB, the F flag will be set if the current assembly XMODE CCW setting is CCW1, otherwise omitted. Ignored if the ORB is not provided by the IOCB macro.
IRB	none	Specifies the address of the area used for storing of the Interruption-Response Block (IRB). If both SCHIB= and IRB= are omitted, an area is reserved by the IOCB macro for storing of both structures. Otherwise, specify 0 to provide the IRB storage area during execution.
KEY	0	Storage access key used by the subchannel in the embedded ORB for the next usage of RAWIO macro. Ignored if the ORB is not provided by the IOCB macro.
ORB	none	Specifies the address of the Operation-Request Block (ORB) used by the next RAWIO macro. Specify 0 if the ORB is provided during execution. The externally supplied ORB must provide its own values for its CCW=, FLAG=, and KEY= parameters. If omitted, an ORB is provided by the IOCB macro using its own CCW=, FLAG=, and KEY= parameters.
SCHIB	none	Specifies the address of the area used for storing the Subchannel-Information Block (SCHIB). If both SCHIB= and IRB= are omitted, an area is reserved by the IOCB macro for storing of both structures. Otherwise, specify 0 to provide the SCHIB storage area during execution.

Programming Notes:

A single IOCB must not be shared between devices. Each device requires its own IOCB structure.

IOCBDS

Source file: macros/IOCBDS.mac

Macro Format:

```
IOCBDS
```

The IOCBDS macro maps the fields of the IOCB to the DSECT IOCB.

Assembly Considerations: None

Execution Considerations:

- All sync macros that use the IOCB structure, must have addressability to the IOCB DSECT via an assembler USING directive for the IOCB structure associated with the targeted device.

Label Field Usage: Prohibited

Positional Parameters: None

Keyword Parameters: None

IOINIT

Source file: macros/IOINIT.mac

Macro Format:

```
[label]   IOINIT [MASK=hex]
```

The IOINIT mask initializes the required control register to enable input/output interruptions for requested parallel channels or subchannel interruption subclasses.

Assembly Considerations:

- A preceding ARCHLVL macro is required.
- A local base register must be established before IOINIT.

Execution Considerations: None

Label Field Usage:

The label field, if present, defines an assembly label associated with the first instruction generated by the macro.

Positional Parameters: None

Keyword Parameters:

Parameters Used by Parallel Channel Operation (Levels 1-4)

Keyword	Default	Description
MASK	none	Up to 8 hexadecimal digits identifying the parallel channels enabled for interruptions. If omitted, all channels are enabled. Do not defined the digits as a hexadecimal self-defining term.

Parameters Used by Channel Subsystem Operation (Levels 5-9)

Keyword	Default	Description
MASK	none	Up to 2 hexadecimal digits identifying the subchannel interruption subclasses enabled for interruptions. If omitted, all interruption subclasses are enabled. Do not define the digits as a hexadecimal self-defining term.

IOTRFR

Source file: macros/IOTRFR.mac

Macro Format:

```
[label]   IOTRFR reg
```

The IOTRFR macro calculates the number of bytes transferred by the last CCW of the channel program following a successful RAWIO operation.

Assembly Considerations:

- A preceding ARCHLVL macro is required.
- The IOCB DSECT must be defined preceding the IOTRFR macro.
- Addressing of the device's IOCB structure must be established through the IOCB DSECT preceding the IOTRFR macro.
- A local base register must be established preceding IOTRFR.
- The current assembly XMODE CCW setting controls how the calculation is performed.

Execution Considerations: None

Label Field Usage:

The label field, if present, defines an assembly label associated with the first instruction generated by the macro.

Positional Parameters:

1. The required reg parameter identifies the register in which the calculated transfer count is supplied.

Keyword Parameters: None

RAWAIT

Source file: macros/RAWAIT.mac

Macro Format:

```
[label]   RAWAIT reg
                ,FAIL=label
                [,CHAN=[X'FE'|n]]
```

The RAWAIT macro waits for an expected unsolicited interruption from the device identified by the addressed IOCB structure. The expected unit interruption condition is identified by the IOCB WAIT= parameter. Once the expected interruption occurs from the device, control passes to the next sequential instruction following RAWAIT.

Receiving any channel status or receiving an unexpected unit status causes the waiting operation to fail. For the channel subsystem, the waiting operation may additionally fail if the device is not operational, an interruption occurs without status, or the interruption is not unsolicited (meaning an operation is occurring via a RAWIO operation).

Assembly Considerations:

- Preceding ARCHLVL and ARCHIND macros are required.
- The IOCB DSECT must be defined preceding the RAWAIT macro.
- Addressing of the device's IOCB structure must be established through the IOCB DSECT preceding the RAWAIT macro.
- Assigned storage area addressing via general register 0 preceding RAWAIT.
- Required input/output structures: CSW, or IRB and SCSW
- A local base register must have been established preceding RAWAIT.

Execution Considerations:

- Interruptions from devices other than the IOCB specified device are ignored and lost, if the unexpected device's channel or interruption subclass is enabled for interruptions.
- Channel subsystem uses general register 1 for subchannel identification in assembly architecture levels 5-9.
- RAWAIT must not be used to wait for completion of an input/output operation initiated by RAWIO. RAWIO performs its own waiting operation.

Label Field Usage:

The label field, if present, defines an assembly label associated with the first instruction generated by the macro.

Positional Parameters:

1. The positional reg parameter identifies a general register other than 0 or 1 for channel subsystem structure addressing. Required by assembly architecture levels 5-9.

Keyword Parameters:

Keyword	Default	Description
CHAN	X'FE'	PSW channel interruption mask used by assembly architecture levels 1 and 2. Used internally for CPUWAIT macro. Ignored by other architecture levels.
FAIL	none	This keyword parameter is required. Control passes to this label if • any channel status or unexpected unit status occurs, • the subchannel has become inoperable, • the subchannel status was not pending, or • the subchannel was active (presumably by a RAWIO macro).

RAWIO

Source file: macros/RAWIO.mac

Macro Format:

```
[label]   RAWIO reg,
                ,FAIL=label
                [,CERR=label]
                [,UERR=label]
                [,SENSE=label]
                [,EOF=label]
                [,CHAN=[X'FE'|n]]
```

The RAWIO macro initiates an input/output operation with the device identified by the addressed IOCB structure and waits for its completion by the device.

Multiple interruptions may occur during the operation of a single input/output operation. Status is accumulated and the operation continues to wait until channel end and device end are both present for parallel channels (architecture levels 1-4) or primary status is present for a subchannel (architecture levels 5-9).

Upon completion of status accumulation, the specified conditions, in the order identified in the macro format, are checked and control is passed if detected. If none of the **requested** conditions are detected, control passes to the next sequential instruction following the RAWIO macro and the program must inspect the IOCB itself to determine further actions required by the program for any possible status conditions not detected by the IOCB CERR= or IOCB UERR= masks.

Refer to the “HOWTO Perform Input/Output Operations” section for more details concerning program interaction with the IOCB structure and the sync macros.

Assembly Considerations:

- Preceding ARCHLVL and ARCHIND macros are required.
- The IOCB DSECT must be defined preceding the RAWIO macro.
- Addressing of the device's IOCB structure must be established through the IOCB DSECT preceding the RAWIO macro.
- Assigned storage area addressing via general register 0 preceding RAWIO.
- Required input/output structures: CSW, or IRB and SCSW, CCW0, CCW1 and SCHIB,
- A local base register must have been established preceding RAWIO.

Execution Considerations:

- Interruptions from devices other than the IOCB specified device are ignored and lost, if

the unexpected device's channel or interruption subclass is enabled for interruptions.

- Channel subsystem uses general register 1 for subchannel identification in assembly architecture levels 5-9.

Label Field Usage:

The label field, if present, defines an assembly label associated with the first instruction generated by the macro.

Positional Parameters:

1. The positional reg parameter identifies a general register other than 0 or 1 for channel subsystem structure addressing. Required by assembly architecture levels 5-9.

Keyword Parameters:

Keyword	Default	Description
CERR	none	When specified, completion channel status is inspected against the IOCB CERR= mask. If any conditions in the mask are set, control is passed to the supplied label.
CHAN	X'FE'	Channel PSW interruption mask used by assembly architecture levels 1 and 2. Used internally for CPUWAIT macro. Ignored by other architecture levels.
EOF	none	Some devices supply a unit exception status when a physical end-of-file condition exists, for example, a card reader, tape, or count-key-data devices. For such devices, when specified, unit status is checked for a unit exception. If unit exception status is present, control passes to the supplied label. When unit exception status is specified in the IOCB UERR= mask and the RAWIO UERR= label has been supplied, the UERR= label takes precedence over this parameter.
FAIL	none	This keyword parameter is required. Control passes to this label if • the input/output operation fails to be initiated, or • if a subchannel has presented primary status but either channel end or device end status is not present.
SENSE	none	When specified, unit status is checked for unit check. If unit check status is present, control passes to supplied label. When unit check status is specified in the IOCB UERR= mask and the RAWIO UERR= label has been supplied, the RAWIO UERR= label takes precedence over this parameter.
UERR	none	When specified, completion unit status is inspected against the IOCB UERR= mask. If any conditions in the mask are set, control is passed to the supplied label.

Programming Notes:

Multiple interruptions may occur during the operation of a single input/output operation. Status is accumulated until channel end and device end are both present for parallel channels (architecture levels 1-4) or primary status is present for a subchannel (architecture levels 5-9).

When control passes to the SENSE= label and concurrent sense is available, the stored IRB contains the concurrent sense information. If parallel channels are in use, the program must solicit the sense information from the control unit through a separate I/O operation.

No facility is provided for the detection of a missing input/output interruption. In this case, the program may wait forever. Manually initiating an interrupt for the device may free the program.

HOWTO Simply Wait

The CPU is primarily in one of two states:

- stop, or
- run

Because in the stop state the CPU is doing nothing, it can be of no interest to a program.

For a program the only CPU state of interest is the run state. In the run state four cases are of interest. Depending upon whether an operating system is present or only a bare-metal program is resident, the four cases imply different things.

Case	Interruptions Enabled	Instructions Executing	SATK Bare-metal Program	Operating System
1	yes	yes	ISR required	Normal OS case
2	yes	no	CPUWAIT	All active work waiting for events
3	no	yes	Normal program case	ISR running
4	no	no	DWAIT or DWAITEND	Catastrophic failure

The SATK bare-metal program operates exactly opposite that of an operating system. The normal situation of the bare-metal program is to execute instructions in case 3. Whereas the normal situation for an operating system is to execute instructions in case 1.

As this table indicates, when a program uses the CPUWAIT macro and the program is executing instructions but not expecting an event, it is expected that interruptions are disabled. At the start of CPUWAIT, the program is in case 3. Internally, CPUWAIT moves to case 2 while it is actually waiting. CPUWAIT restores that condition when it recognizes the requested event, internally moving from case 2 to case 3 before passing control to the program.

On occasion a program without the complexity of interrupt service routines may need to wait for specific events expected by the program. These events can be:

- expired timers,
- input/output operations, and
- other event situations.

If the program does not have any interrupt service routines, how can this be accomplished? At the level of the interruption an interrupt service routine is nothing more than a place to which control is passed to handle the interrupt. Because such events can occur asynchronously from the running program, interrupt service routines typically require saving upon entry and a restoration of the program's state upon exit. By causing the interruption to pass control to a place where the program's state is known, only the passing of control is required.

If a program expects to wait, the place at which its state is known is at the point in the program where it expects to perform the wait. In other words, inline, is the best place to perform the wait. The CPUWAIT macro performs an inline wait. It does this by:

1. establishing a new PSW for the interrupt class upon which it is waiting that passes control following the instructions, and optionally saving the previous new PSW;
2. loading an **enabled** wait PSW for the anticipated event (this actually does the waiting; and
3. receiving program control following the event
4. passing control as specified by the class' related parameter, optionally restoring the previous new PSW before doing so.

The need to save program state is eliminated because the program ceases instruction execution. The formal interrupt service routine is eliminated because the instructions following the CPUWAIT macro process the event.

Two events are supported by the very low level CPUWAIT macro:

- External interruptions and/or
- Input/Output interruptions.

Both are supported by the same CPUWAIT macro depending upon its parameters. This allows for example, with the appropriate preparation by the program to wait for an input/output event or a timeout if the input/output event doesn't occur.

In either case the program continues execution following the CPUWAIT macro. The CPUWAIT macro passes control as specified by its parameters. It also allows the program to differentiate between the interruption classes by setting the condition code:

- 1 – an external interruption occurred, or
- 2 – an input/output interruption occurred.

The program is required to examine any interruption information supplied in assigned storage locations related to the interruption class and proceed accordingly.

HOWTO Perform Input/Output Operations

Mainframe systems from their inception are designed to perform input/output operations in an interruption driven environment. Interruptions result from initiated input/output operations and other input/output related notifications occur through interruptions. While it is technically possible to perform input/output operations without utilizing interruptions, it is not possible to be informed of input/output related notifications without interruptions. The use of the input/output system without interruptions has never been encouraged. SATK therefore only supports the input/output system utilizing interruptions.

Interruptions resulting from initiated input/output operations are referred to as “solicited” interruptions. Event notification interruptions outside of input/output operations are referred to as “unsolicited” interruptions.

Mainframes have utilized two different architectures for the input/output system. Both are based upon the concepts of channel programs executed by the channel with the aid of the devices themselves. From the view of the CPU, Initiating and termination of the channel programs differs between the two architectures. The form and structure of input/output notifications also differ. Providing a simple, consistent interface to the two different I/O architectures without losing flexibility is the primary goal of the supplied macros.

The first architecture, used individual channels controlled by the program for input/output operations. By the third generation of mainframes, this system was replaced by a channel subsystem that managed all of the physical channels and performed input/output operations on behalf of the program. As the technology advanced, the information necessary for the channel subsystem to perform its tasks increased. New storage based structures were introduced for this purpose.

The macros have no knowledge of the operations, channel commands or details of successful or failed activities. The program (and developer) must supply such knowledge. Under either architecture, the macros will **initiate** an operation and **detect** its success or failure as defined by the program under either architecture. In very simple settings, the input/output operations can be transferred between the two input/output architectures without program changes. See the `hello.asm` sample program for an example of this usage. Because facilities differ between the two architectures, this transportability is not generally likely for complex operations.

The input/output operations typically involve three aspects of the input/output architecture:

- preparing the CPU for input/output operations,
- making individual devices ready for use and then
- using the device for operations.

Seven macros work together for these functions:

- `IOINIT` – prepares the CPU for input/output operations, typically used once by the program;

- IOCB – the assembled structure used to define an input/output device used by the program, used once for each device;
- IOCBDS – creates a DSECT named IOCB, defining the IOCB structure fields, used once by the program;
- ENADEV – makes devices defined by an IOCB macro ready for use by the CPU;
- RAWIO – performs an inline input/output operation with an IOCB defined device; and
- IOTRFR – calculate the number of bytes actually transferred following a successful I/O operation.
- RAWAIT – performs an inline wait for an unsolicited interruption from the IOCB defined device.

All, except IOINIT utilize a common structure in performing these functions.

The generated IOCB structure links the program, the input/output system and actions with the device. Because the ENADEV, RAWAIT, and RAWIO macros are strictly driven through the IOCB a single use of the these two macros can be used for all devices and all input/output operations by simply addressing a different IOCB before executing the inline instructions. This is intended to allow these three macro to be placed in a subroutine and reused by the program.

Definition	Creation	Init/Term	Process
IOCBDS	IOCB	Init: IOINIT Init: ENADEV	M: IOTRFR M: RAWIO M: RAWAIT

All of the macros, with the exception of IOCBDS, are dependent upon the current architecture level implied by the current XMODE PSW setting. The architecture level is used to determine which I/O architecture scheme is in use. Channel command words can utilize either of two formats. Only the IOTRFR macro is sensitive to the CCW format specified by the XMODE CCW setting.

IOCB Structure

The IOCB structure links all of the macros (with the exception of the IOINIT macro) to the program and each other. The IOCB is the mechanism by which a SATK program and the input/output related macros communicate with each other and attached devices. The IOCB also normalizes the differences between the two input/output architectures. The IOCBDS macro identifies certain fields within the structure as read-only. The program should not alter these fields. They are critical to the operation of the macros.

The two input/output architectures both depend upon assigned storage areas and the use of

channel-command words for communication with a device performing an input/output operations. Each uses a similar, although not identical, structure for communicating the results of an operation. Locating this information differs, but its location is made transparent to the SATK program through use of the IOCB structure.

The channel subsystem specific structures are not inherently part of the IOCB structure, although the IOCB macro can as a convenience create the Operation-Request Block (ORB) and reserve space for the transient needs of the program for access Subchannel-Information Block (SCHIB) and Interrupt-Response Block (IRB) access. Even when the IOCB provides these structures, they are external to the IOCB structure. The addresses for each are used by the I/O macros as if the structures were created separate from the IOCB. Whether a field is read-only or not, the information within the IOCB structure is shared between the I/O macros and the program.

The IOCB structure is assembled by the `IOCB` macro and defined by the `IOCBDS` macro. All fields may be examined by the program, but only specific fields should be modified by the program. These fields are identified by an '`X`' in the IOCB dummy section created by `IOCBDS`. Intentionally the IOCB structure may be completely specified during assembly or completely created during program execution or some combination. Regardless of the creation, each device using the I/O macros described in this section requires an IOCB structure.

The IOCB is utilized for multiple purposes related to the input/output system. Some fields are expected to be modified by the program as needed. These

- Provide the logical identification of the corresponding device or subchannel as known by the program, field `IOCBDEV`;
- Identify the input/output operations to be performed, fields `IOCBCAW` or `IOCBORB`;
- Specify conditions related to the device or subchannel operation representing errors, of special significance or completely ignored, fields `IOCBUM`, `IOCBCM`, and `IOCBWAIT`; and
- Locate structures required by the channel subsystem used by the `ENADEV`, `RAWIO` or `RAWAIT` macros, fields `IOCBIRB` and `IOCBSIB`.

Other fields, are expected to only be examined by the program, but not modified. These fields:

- Provide the corresponding I/O system designation for the device or subchannel when interacting with it, initialized by `ENADEV`, field `IOCBID`;
- Normalize I/O architecture specific interruption information into the IOCB structure for examination by the `IOTRFR`, `RAWIO` or `RAWAIT` macros, fields `IOCBSCCW`, `IOCBSCNT`, `IOCBUS`, and, `IOCBCS`; and
- Internally used by `RAWIO` or `RAWAIT`, fields `IOCBZERO`, `IOCBUT`, `IOCBCT`, `IOCBSC`.

The following table summarizes the relationships between the various macros, the IOCB fields and the program. The column heading 'CH' indicates the field is used by channel-

based input/output system. The column heading 'CS' indicates that the field is used by the channel subsystem.

IOCB Field	Program	CH	CS	IOCB Parm.	ENADEV	RAWIO	IOTRFR	RAWAIT
IOCBID	RO	X	X		initializes	input		input
IOCBDEV	RW	X	X	DEV	input			
IOCBUM	RW	X	X	UERR		input		
IOCBCM	RW	X	X	CERR		input		
IOCBUS	RO	X	X			updates		updates
IOCBCS	RO	X	X			updates		updates
IOCBUT	RO	X	X			updates		updates
IOCBCT	RO	X	X			updates		
IOCBSC	RO		X					updates
IOCBWAIT	RW	X	X	WAIT				input
IOCBSCCW	RO	X	X			updates	input	
IOCBSCNT	RO	X	X			updates	input	
IOCBCAW	RW	X	***	CCW, KEY *		input		
IOCBORB	RW	***	X	ORB		input		
IOCBIRB	RW		X	IRB		input **		input **
IOCBSIB	RW		X	SCHIB	input **			

* IOCB macro parameters CCW, KEY and FLAG are used to initialize the embedded ORB when an explicit ORB parameter is omitted.

** The field is not altered by the macro. However, the area to which it points is updated by the macro.

*** The IOCBCAW and IOCBORB fields overlap or correspond. Both fields serve to identify to the I/O system the location of the channel program constituting the initiated input/output operation depending upon the system in use. The IOCBCAW directly locates the channel program. The IOCBORB field indirectly locates the channel program by providing the location of the ORB that contains the explicit location.

Using the Input/Output Macros

This section describes the steps required in a program to use the basic input/output macros. The following two sections give details related to the use of the macros in a specific input/output architecture.

The RAWIO and RAWAIT macros depend upon detecting a solicited or unsolicited interruption, respectively, from the targeted device. Interruptions from any other device are ignored and lost. See the input/output architecture considerations concerning the IOINIT macro for details on managing interruptions.

Step 1 – Define the Input/Output Structures to the Assembly

Assemble the input/output related dummy sections:

- DSECT macro omitting the NAME parameter, selecting architecture sensitive dummy sections,
- DSECT NAME=(CH,IOCB), for a channel I/O system,
- DSECT NAME=(CS,IOCB), for a I/O channel subsystem, or
- DSECT NAME=IOCB, equivalent to using the IOCBDS macro itself.

Step 2 – Assemble Device Definitions

Use the IOCB macro to define the devices the program plans to use. IOCB definitions are static. The macros described here do not perform device discovery involving the use of the SENSE ID channel command. There is nothing stopping the program from using a dynamically altered IOCB for that purpose.

Device definitions may be placed anywhere in the assembly. Each macro that uses the IOCB structure expects to address the contents via a USING directive for the IOCB dummy section.

If the channel program consists of the same input/output operations it may be assembled with the device definition.

Step 3 – Prepare the CPU for Input/Output Operations

Use the IOINIT macro to prepare the CPU to accept interruptions from various sources. The IOINIT macros loads the appropriate control register for this purpose. By default, all input/output channels or subchannels are enabled for interruption recognition by the CPU.

Step 4 – Enable Each Device for Input/Output Operations

For each device, use the ENADEV macro to prepare the device for use. By incorporating the ENADEV macro in a subroutine, a single use of the macro may be used for each device by simply loading the register used to address the IOCB dummy section with each IOCB's address.

Step 5 – Perform an Input/Output Operation

The RAWIO macro initiates the execution of a channel program by a device or subchannel

using the IOCB structure associated with the device. It performs the operations and accumulated solicited interruption information for analysis of its success or failure. Special handling of a physical end-of-file condition (EOF parameter) or need to access device sense data (SENSE parameter) are provided. If either are used the unit exception or unit check conditions, respectively, must not be set in the IOCB IOCBUM field or IOCB macro UERR parameter.

Depending upon the needs of the program the IOCB fields may be updated for the specific operation. Different operations with a device or subchannel may require different handling of the status indicators. Some may be errors in some settings but not others. Program alteration of the IOCBUM or IOCBCM fields may be required. Correspondingly, a different sequence of input/output operations may require changes to the IOCB or channel subsystem structures to utilize a different channel program by alteration of the IOCBCAW or the address of the first channel command word in the channel subsystem Operation-Request Block.

For input/output operations where the number of bytes transferred is unpredictable, the IOTRFR macro calculates the number of bytes actually transferred based upon the accumulated status information in the IOCB. The macro assumes a successful completion of the input/output operation. This usually requires suppression of incorrect length indication. Non-default IOCBUM, CCW and ORB flags may be required to properly utilize the indication.

Step 6 – Detect Device Attention or Other Event

Some devices utilize the attention unit status to indicate that input is available. This is usually the case for devices used by an operator, for example a console or graphic display unit. Use the RAWAIT macro to perform this function. The macro waits for an unsolicited interruption from the device with any of the status indications set by the IOCB macro WAIT parameter or contained in the IOCBWAIT IOCB field at the time the RAWAIT macro is executed by the program.

By default, the IOCB WAIT parameter sets the attention status.

Channel Input/Output Considerations

No special considerations exist for use of the IOCBDS, ENADEV, or IOTRFR macros.

IOCB

The IOCB positional macro parameter DEV specifies a device's channel and unit address.

The address assigned to channel attached device when using the Hercules emulator is the address provided in the device's configuration statement.

IOINIT

Control register 2 is loaded by the IOINIT macro on System/370 or System/380 systems.

The CPU reset that is part of the IPL function will set all bits in control register 2 to ones. If all channels are to be enabled for interruptions, then IOINIT is superfluous following an IPL.

System/360 systems do not support control registers and on these systems, IOINIT generates no instructions.

Managing I/O Interrupts on System/370 in EC mode or System/380

To ensure interruptions are not lost though, a channel must not be enabled for interruptions if the program is not prepared to lose the interruption. At the level of an interruption, the CPU is unable to differentiate between solicited or unsolicited interruptions. If the CPU is enabled for input/output interruptions by the PSW, and an interruption occurs from a device that is on a channel that is enabled for interruptions, the CPU will accept it. If the channel is not enabled for interruptions, the interruption will remain pending until such time as it is enabled.

By separating a device or set of devices onto its own channel, the program can control which devices have the potential for lose. Devices used exclusively with RAWIO, can generally be safely used without concern. Unsolicited interruptions though can occur at any time in relationship to other operations. Hence devices which may require use of the RAWAIT macro should be separated. In this setting IOINIT can manage which channels can present an interrupt and which can not. Before issuing the RAWAIT macro, IOINIT can enable the channel on which the device resides and disable the others. Before issuing the RAWIO macro, IOINIT can be used to disable interruptions from these devices and enable them for the other devices used for normal operations. In the case of a solicited operation following the detection of an unsolicited event, no change in enabled channels is required.

More advanced management of interruptions requires formal interrupt service routines.

Managing I/O Interrupts on System/370 in BC Mode

The same considerations apply to System/370 in BC mode. However, the use of channel masks in control register 2 only apply to channels 6 or above. For channels 0 through 5 inclusive the techniques described in the following section apply.

Managing I/O Interrupts on System/360

The same considerations apply to System/360 as on System/370. However, the “knob” that controls channel interrupts is within the PSW itself. The PSW enabling input/output interruptions is embedded within the CPUWAIT macro used by the RAWIO or RAWAIT macro. Run-time modification of the mask within this waiting PSW is not readily available. In this case separate RAWIO or RAWAIT macros should be used with the appropriate enabling channel mask set by the CHAN parameter available on the two macros.

RAWIO

Input/output operations are initiated at the device by means of a START IO instruction. The

CPU then waits for solicited interruptions until the accumulated status provided by the CSW indicates the channel end and device end status.

Any interruption that occurs for the target device is considered a solicited interruption by the RAWIO macro. No mechanism exists to differentiate between solicited and unsolicited interruptions for channel attached devices. Only the state of the program knows which is expected.

RAWAIT

Any interruption that occurs for the target device is considered an unsolicited interruption by the RAWAIT macro. No mechanism exists to differentiate between solicited and unsolicited interruptions for channel attached devices. Only the state of the program knows which is expected.

Multiple status conditions are possible for unsolicited interruptions besides the attention unit status.

Channel Subsystem Considerations

No special considerations exist for use of the IOCBDS or IOTRFR macros. While simple scenarios may not require an understanding of the roles of the channel subsystem specific structures, generally programmer familiarity will be required.

IOCB

The IOCB positional parameter DEV specifies a subchannel's device number. The device number is analogous to a channel attached device's channel and unit addresses.

The subchannel device number when using the Hercules emulator is the address provided in the device's configuration statement.

The areas identified by the SCHIB or IRB parameters must be large enough to contain the structure that may be stored. For the SCHIB, this is 13 full words, 52 bytes, full word aligned. For the IRB this must be 24 full words, 96 bytes, full word aligned. It is possible for all IOCB's to share the same 96-byte area for their respective SCHIB or IRB because at any given point in time the area will be used either for only a single SCHIB (by the ENADEV macro) or IRB (either the RAWIO or RAWAIT macro) related to one subchannel.

IOINIT

Control register 6 is loaded by the IOINIT macro on systems having a channel subsystem with a mask enabling all subchannel subclasses for interruptions. Control register 6 is set to all zeros by the CPU reset that occurs during the IPL function. Without a setting of mask bits in control register 6, no subchannel interruptions will be recognized by the CPU. IOINIT is required when a channel subsystem is in use.

Managing Subchannel Interruptions

The potential for interruption lose exists with channel subsystem interruptions in the same way with channel attached devices. Because the channel subsystem takes over responsibility for management of the channels, interruptions are managed through the concept of an interruption subclass. When the I/O reset associated with the IPL function occurs, all subchannels are placed in interruption subclass 0. Interruption subclasses are limited to eight, subclasses 0 through 7 inclusive. By adjusting the enabled interruption subclass in control register 6, the program can control which devices are enabled at any given time. By disabling an interruption subclass, all subchannel assigned to the subclass remain interruption pending until the CPU is enabled for input/output interruptions and the interruption subclass is enabled by its mask bit being set to one in control register 6. As with channel attached devices, the interruption mask can be manipulated by the IOINIT macro.

See the ENADEV considerations for setting a subchannel's interruption subclass.

ENADEV

The operation of ENADEV on channel subsystem is more complex than for channel attached devices. Each subchannel in the channel subsystem is examined by using the STORE SUBCHANNEL instruction until the device number, IOCB IOCBDEV field, is found or until all subchannels have been examined. Once the subchannel with the device number is located, the IOCBDID IOCB field is initialized with the subchannel's subsystem identification number. The subchannel itself is enabled with the channel subsystem using a MODIFY SUBCHANNEL instruction.

By default the subchannel will utilize an interruption subclass of zero. The interruption subclass can be modified when the subchannel is enabled by supplying a value in the register indetified by the ENADEV CLS= parameter. If the CLS= parameter is omitted, the interruption subclass remains unchanged. See the section “Managing Subchannel Interruptions” why this may be useful.

By specifying the CSNS=YES parameter, the subchannels enabled by the macro will be enabled for concurrent sense. Separate macros must be use to enable a mixture of some subchannels enabled for concurrent sense and some that are not.

The strategy used by the ENADEV macro in locating the subchannel associated with a device number is not encouraged for many devices. Rather than searching the subchannels for each desired device number, it is more efficient to scan the subchannels once and repetitively scan a memory resident table containing the device numbers sought by the program until each is found or the subchannels are exhausted. For a small number of devices this strategy is simpler if less efficient.

RAWIO

Input/output operations are initiated at the subchannel by means of a START SUBCHANNEL instruction. The CPU then waits for solicited interruptions from the subchannel until primary

status is presented by the SCSW stored with the IRB by the TEST SUBCHANNEL instruction. Interruptions occurring subsequent to the START SUBCHANNEL instruction are assumed to be solicited by the RAWIO macro.

Depending upon the nature of the input/output operations, a program supplied Operation-Request Block, ORB, may be required. The ORB macro may be used to assemble the structure.

If the program elects to use different channel programs with the subchannel with which an IOCB is associated, the ORB must be updated by the program before the RAWIO macro is used regardless of whether it is supplied by the program or supplied by the RAWIO macro.

Use of concurrent sense is possible with the RAWIO macro provided the subchannel has been enabled for it through use of the ENADEV CSNS=YES parameter. The location to which control is passed for unit check device status must be prepared to examine the IRB, located by the IOCB IOCBIRB field, for concurrent sense. The assumption is that the sense information is available in the stored IRB. The SCSW bit 14 and the ERW bit 7 must both be one for concurrent sense information to be present in the IRB ECW. Detailed formats for areas other than the SCSW must be supplied by the programmer.

RAWAIT

The RAWAIT macro waits for an interruption from the target subchannel. Any interruption from the target subchannel for which the SCSW device active status is present is treated as an error.

Only the attention unit status condition is reflected to the program via an unsolicited interruption. Other status conditions present with channel attached devices are handled directly by the channel subsystem.