EUROGAM Ebc Reference Card

(for version 1.0)

Setting up

There are 2 steps involved in producing an event builder customised to individual experimental requirements.

A data file of EUROGAM event builder source language should be created using any text editor. The data file has to contain sections as described in File Sections below.

Secondly, the complete description file must be passed through the Event Builder Compiler, see elsewhere.

Statement Definitions

The compiler is not case sensitive. Blank lines between statements are ignored. There are restrictions on line breaks in the middle of statements. Each statement should start on a new line. Unnecessary white space is ignored. Comment lines and trailing comments should be indicated with an exclamation mark (**!**).

<...-name> ... upto 8 characters (alphanumeric plus _)

File Sections

input formats ... see Input Definitions
<input-definition-statements>
output formats ... see Output Definitions
<output-definition-statements>
commands ... see Command Definitions
<command-statements>
finish ... File terminator

Input Definitions

This is the first section of information required. It defines the maximum set of data items to be expected in each event to be processed. Each parameter is labelled by an "address".

<item-name> <address>
<group-name> [<group-range>] (<item-list>)

where
<group-range> = one or more group ranges of the form
<integer>
or <integer> **:** <integer>
separated by commas
<item-list>= one or more <item-name>s separated by commas

Output Definitions

Definition of the maximum set of data items to be output from each event processed. There are restrictions on the use of names to ensure consistency with the **input format** definitions.

<item-name> <address>
<group-name> [<group-range>] (<item-list>)

where <group-range> and <item-list> are explained in the input definitions section

Command Definitions

Commands all start on a new line except where indicated. For example, a simple command following an **if** statement.

<expression> ... see Arithmetic Commands
<value> is one of <integer>, <item-name> or <local-variable>
<variable> is <item-name> or <local-variable>

A <local-variable> is any name used in **commands** section not defined in **input** or **output formats** section. Qualification bits may be accessed wherever <item-name> appears as follows:

<item-name>**.q0** or
<item-name>**.q1**

The IF Command

The **if** command allows a statement or block of statements to be executed or not depending on the outcome of some test. Various conditions may be applied to data parameters. Upto 6 nested **if** blocks are allowed.

if <condition> **then** <statements>

where
<condition> = <variable> <test> <value>
or<variable> **passes** [<value>, <value>]
or<variable> **fails** [<value>, <value>]
<statements> = <single-command>
or {
<one-or-more-commands>
}
<test> = one of **lt le eq ge gt ne**

The WHENEVER Command

This allows the processing of the same statement or block of statements on many groupnumbers. The statement(s) are executed for every group belonging to <group-name> that is present in the event. **Whenever** blocks may not be nested, but they may be combined with **if** blocks.

whenever <condition> **then** <statements>

where
<condition> = **group** = <group-number>
or **group** = <group-name>
or **word** = <sitem-name>
<statements> are as given above for the IF command
<sitem-name> is an itemname defined with the single item format in the **input formats** section.

Arithmetic

<variable> = <expression>
increment <variable>
decrement <variable>

where
<expression> = an expression involving <value>s as operands and the following operators
Arithmetic operators: +, −, *, /
Bitwise operators: **&, ior, xor, not,** <, >

Only integer arithmetic is possible. Upto 6 levels of parenthesis are allowed. C operator precedence and associativity rules are followed.

REJECT Command

Discards the event or the specified part of the event.

reject event
reject group
reject <group-name>
reject <sitem-name>

The **event** keyword rejects the whole of the current event. The **group** keyword is only valid in a **whenever** block and discards the current groupnumber. The third form rejects all groups belonging to <group-name>. The last form rejects the single item named.

SUPPRESS Command

suppress scheme <integer> **file** <file-name>

Does shared suppression according to the instructions laid out in <file-name> for scheme <integer>.

The STREAM Command

Allows the user to output all or part of an event onto different streams.

stream <stream-number> **event** *nosort*
stream <stream-number> <output-list> *nosort*

where

<output-list> =	one or more <group-name>s and/or <sitem-name>s defined in the output definitions section separated by commas
nosort =	optional keyword for output not to be included in the sort

Writing Event Builder Files

An example:

```
! Brief example event builder language source file.
INPUT FORMATS
trigger [255]   (type,mraw)
germ [1:32,34,71] (e1,e2,t,tbd,ebgo,tbgo,hit)
OUTPUT FORMATS
trigger [255]   (type,mraw,nge)
germ [1:32,34,71] (e1,t)
COMMANDS
whenever group = trigger then
if Mraw lt 8 reject event
if type ne 1 reject event
a=2000
b=15000
c=300
d=10000
e=8000
f=30000
whenever group = germ then
sumenergy = sumenergy + e2 + ebgo
if E1 fails [a,b] then reject event
suppress scheme 1 file /eb/shared-suppr
increment nge
if T1 passes [c,d] then increment FT1
if SUMENER fails [e,f] then reject event
if FT1 lt 1 then reject event
stream 0 event
finish
```

Produced by Linda Pratt and John Cresswell, April 1991 v1.0
Contact PRATT or CRESSWELL @ UK.AC.LIV.PH.GB

For copies of the EUROGAM Event Builder Compiler User Manual ... extract document EDOCxxx from the NSF document server.