EMacros.jcl contains a group of IBM assembler routines commonly used by many of my projects. It is connected as a subproject to many of my projects in GITHUB.

There are 3 macros in this file ESTART, ERETURN, EREGS. These generate the standard routine linkage plus additional "eye catcher" to help locate stuff in dumps. In addition, an option is available to specify an alternate entry point for PL/1 F programs. ERETURN generates a standard compatible return sequence. EREGS defines the symbols R0 to R15 as 0 to 15 for symbolic register reference. Side benefit it the R# will be in the xref.

Upon entry to a module, standard linkage is register 1 points to a list of addresses of parms being passed, with the last one having the high order bit set to 1.
Examples:

```
     R1 -> A(PARM1)                or R1 -> X'80'
           X'80'                            AL3(PARM1)
           AL3(PARM2)
```

PL/I F does not follow the standard linkage conventions. Register 1 points to a list of addresses of DOPE vectors for each parm. Each DOPE vector starts with the address of the data. Therefore, extra code is needed after the ESTART macro to map make a standard address list.
Examples:

```
     R1 -> A(DOPE1) -> A(PARM1)         or R1 -> X'80'
           X'80'                           AL3(DOPE1) -> A(PARM1)
           AL3(DOPE2) - > A(PARM2)
```

PL/1 F had many ways to set up parms. The methods described here only works if the parms are static structure names.
Example:

```
      DECLARE 1 SOMEPARM_1  STATIC,
                2 SOMEPARM  ....;
         ...
      CALL SOMEPGM(SOMEPARM_1);
```

The ESTART macro model is:

```
&NAME    ESTART &TYPE=CSECT,&DESC=,&VER=,&BASE=,&REGS=NO,              *
               &PLIF=
Where:
```

- NAME is the CSECT name of the module.
- DESC is a description of the module. mostly an eye catcher for dumps.
- VER is the version. also used for debugging purposes.
- BASE defines a base registed for the CSECT
- REGS=YES means you want the symbolic R0-R15 generated for registers.
- PLIF if specified, causes an alternate entry point for PL/1 F programs to use.

Examples:

```
MYPGM1  ESTART DESC='SOME COMMENTS',VER='1.2.3',BASE=12,REGS=YES
MYPGM2  ESTART DESC='SOME COMMENTS',VER='1.2.3',BASE=12,REGS=YES,    *
               PLIF=MYPGMP
```

MYPGM1 generates a CSECT called MYPGM1 with the description and version info given. In addition, 12 is established as the base register, a save area is created and the registers save per standards.
MYPGM2 generates a CSECT called MYPGM2 with the description and version info given. In addition, 12 is established as the base register, a save area is created and the registers save per standards. Note the '*' should be in column 72. A alternate entry point MYPGMP is created.

To utilize the PL1F interface, immediately after the ESTART macro you code:

```
MYPGM3   ESTART ...,PLIF=PL1CALL,REGS=YES
         BNE   NOTPL1
         LM   R2,R5,0(R1)          this code assumes 4 parms passed
         L    R2,0(,R2)            get the addr from DOPE vector.
         ST   R2,FAKEPARM
         L    R3,0(,R3)            get the addr from DOPE vector.
         ST   R3,FAKEPARM+4
         L    R4,0(,R4)            get the addr from DOPE vector.
         ST   R4,FAKEPARM+8
         L    R5,0(,R5)            get the addr from DOPE vector.
         ST   R5,FAKEPARM+12
         OI   FAKEPARM+12,x'80'    set end of list
         LA   R1,FAKEPARM          SET R1 to FAKEPARM
         B    NOTPL1
FAKEPARM DS  4F
NOTPL1   EQU  *
```

===