05/25/99

I. SYSTEM CALLS

1. Button Parameters -

For all of the button parameters (parameters ending with a B, such as PLAYB) simply pass the button number of the panel that should activate that function. If a particular button does not exist, pass a 0 in its place. For example, a U-Matic is controlled by only five buttons. If the FFWD button is pressed, the call will see if there is a SFWD button. If there is, then the deck will send the FFWD command. If there is not a SFWD button, the call will see if it is in PLAY or not to determine if it should send the FFWD or the SFWD command. All decks with transport controls will work similarly. The SLD calls work like this too; pass a 0 for POFFB if a single power button exists, and a 0 in FOUTB if only one alternating focus button exists.

2. Time Outs -

Several transport system calls have built-in timeouts (VCR1_PAUSE_TO_STOP, etc.) that stop the deck in various conditions. If you do not want the deck to timeout and stop, then override the default and set the timeout value to 0. The maximum value for any timeout is 65535 or about 109 minutes.
Example of a timeout: Cassette decks controlled by system call CAS1 to go to a stop condition after pausing for 10 minutes, and this is not desired in a particular application. In the actual system call is the following statement:

```
CAS1_PAUSE_TO_STOP=6000
```

(*Note* 10 minutes = 6000 for AXCESS programming purposes, as 1/10 second = 1). The following needs to be defined in the DEFINE_CONSTANT area of the program which is being written:

```
CAS1_PAUSE_TO_STOP=0
```

This overrides the information contained within the system call.

3. FIRST Parameter -

The FIRST parameter is used as an offset for various calls. For example, if PLAY is located in position 4 (as possible in an AXCENT) instead of position 1, then pass 4 in the FIRST position. If no offset is needed, pass 0. Even if PLAY is in a different position, the order of the functions must be maintained (STOP=5, PAUSE=6, FFWD=7, etc.) for the calls to work. If the feedback channels need to be moved, then pass the desired first channel in the high byte of FIRST. If PLAY_FB is not needed at 241 but at 109, then pass 109*$100 in the FIRST position. The multiplication by $100 shifts the offset into the high byte. Example: Say you are using an IR controlled receiver that has controls for both a CD player and a cassette deck. The CD controls are in the standard positions, but the cassette controls begin at position 43. The following two lines of AXCESS code

could be used in this situation:

```
  SYSTEM_CALL 'CDP2' (RECEIVER,TP,11,12,13,14,15,16,17,0)       (*CD*)
  SYSTEM_CALL 'CAS3' (RECEIVER,TP,21,22,23,24,25,0,231*$100+43) (*CAS*)
```

The CD call works normally; the 0 in the FIRST position does not change any defaults (actually, sending 241*$100+1 in the FIRST position achieves the same thing). The CAS call changes the offsets so PLAY is in the correct position (in this case 43) and moves the feedback to start at channel 231. This will let the status of both devices to be kept on the same card! NOTE: At the present date, the above example will not work as described because of a feature of the IR/S cards, but the theory is correct. Any feedback below position 241 on an IR card will not be properly saved. This will hopefully be lowered to something like 200 soon, but for now you are out of luck. This system call feature will work on every other control card/device AMX makes.

For screen calls, the FIRST parameter controls the screen run time. If you need the screen to run for 10 seconds, set the FIRST parameter to 10. (NOTE: this parameter is interpreted as seconds, not 1/10's of a second). You can specify different UP and DOWN times by setting the low byte of FIRST to the down time and the high byte of FIRST to the up time. For example, say you have a screen which needs the relay held for 10 seconds to travel down fully and 15 seconds for the screen to travel up fully. You would specify the FIRST parameter to the screen call as: 15*$100+10.
If the FIRST parameter is 0, the default time is 1/2 second, a standard PULSE time.

4. Hardware Requirements -

If you are using a Master card (or AXcent) running software revision 3.29 or lower, some of the calls will not work properly. Older Masters can not use the keywords MIN_TO or GET_PULSE_TIME, which are heavily used by the system calls. If you are using an old Master, include the following line in your code:

```
  #DEFINE MASTER_BELOW_V330
```

This will change all MIN_TO statements to TO statements, and assumes the current pulse time is set to 5 (the default). If you have a version of AXCESS that will not compile the system calls using MIN_TO, get an updated version of AXCESS, as other bugs were fixed dealing with system calls in the compiler that you need.

As of 4/10/98, system calls are now Time-Stamped to indicate revision number. Using MS-DOS or Windows Explorer, you check the revision number of the system call by using the date and time. The date will reflect the date the call was released and the time will reflect the revision. For example, if the file's time was 2:01 AM, the it is revision 2.01. All calls below are noted with current revision available.

II. Generic Button Calls ----------

Version 2.0

A. Button_5

SYSTEM_CALL 'BUTTON_5'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)

Provides MIN_TO functionality with no feedback for decks with
5 button hand controls (PLAY,STOP,PAUSE,FFWD,REW,RECORD)
The SFWDB and SREVB do not do anything in this call.
The FIRST parameter is only used to specify offset only.
This call is primarily used for DVD's where all functionality
is required, but it can be used on any deck.
Current: Version 2.0

B. Button_7

SYSTEM_CALL 'BUTTON_7'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)

Provides MIN_TO functionality with no feedback for decks with
7 button hand controls (PLAY,STOP,PAUSE,FFWD,REW,SFWD,SREV,RECORD)
The FIRST parameter is only used to specify offset only.
This call is primarily used for DVD's where all functionality
is required, but it can be used on any deck.
Current: Version 2.0

III. CASSETTE calls----------

****
******* Instancing is required on these calls! **********

A. CAS1

```
SYSTEM_CALL [DECK] 'CAS1' (DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,RECB,FIRST)
  searches not available
  sends PLAY (or REC) to exit pause
  sends PLAY & REC together for record
     CAS1_REW_TO_STOP = 1800 (3 min)
     CAS1_PAUSE_TO_STOP = 6000 (10 min)
     CAS1_DEFEAT_FEEDBACK = OFF
     Current: Version 2.0
```

B. CAS2

```
SYSTEM_CALL [DECK] 'CAS2' (DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,RECB,FIRST)
  searches not available
  sends PLAY (or REC) to exit pause
  sends PLAY & REC together for record
  sends STOP for pause
     CAS2_REW_TO_STOP = 1800 (3 min)
     CAS2_PAUSE_TO_STOP = 6000 (10 min)
     CAS2_DEFEAT_FEEDBACK = OFF
     Current: Version 2.0
```

C. CAS3

```
SYSTEM_CALL [DECK] 'CAS3' (DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,RECB,FIRST)
  searches not available
  sends PLAY (or REC) to exit pause
     CAS3_REW_TO_STOP = 1800 (3 min)
     CAS3_PAUSE_TO_STOP = 6000 (10 min)
     CAS3_DEFEAT_FEEDBACK = OFF
     Current: Version 2.0
```

D. CAS4

```
SYSTEM_CALL [DECK] 'CAS4'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
  has discrete searches
  sends PLAY (or REC) to exit pause
     CAS4_REW_TO_STOP = 1800 (3 min)
     CAS4_PAUSE_TO_STOP = 6000 (10 min)
     CAS4_SREV_TO_STOP = 12000 (20 min)
     CAS4_DEFEAT_FEEDBACK = OFF
     Current: Version 2.0
```

E. CAS5

```
SYSTEM_CALL [DECK] 'CAS5'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
  has discrete searches
```

```
  sends PLAY (or REC) to exit pause
  sends PLAY & REC together for record
     CAS5_REW_TO_STOP = 1800 (3 min)
     CAS5_PAUSE_TO_STOP = 6000 (10 min)
     CAS5_DEFEAT_FEEDBACK = OFF
     Current: Version 2.0
```

F. CAS6

```
SYSTEM_CALL [DECK] 'CAS6' (DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,RECB,FIRST)
  searches not available
  sends PAUSE to exit PAUSE
  send PLAY and REC to record
     CAS6_REW_TO_STOP = 1800 (3 min)
     CAS6_PAUSE_TO_STOP = 6000 (10 min)
     CAS6_DEFEAT_FEEDBACK = OFF
     Current: Version 2.0
```

G. CAS7

```
SYSTEM_CALL [DECK] 'CAS7'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
  has discrete searches
  sends PLAY (or REC) to exit pause
  send REC then PLAY for record
     CAS7_REW_TO_STOP = 1800 (3 min)
     CAS7_PAUSE_TO_STOP = 6000 (10 min)
     CAS7_SREV_TO_STOP = 12000 (20 min)
     CAS7_DEFEAT_FEEDBACK = OFF
     Current: Version 2.01
```

IV.CD calls----------

****
******* Instancing is required on these calls! **********

A. CDP1

SYSTEM_CALL [DECK] 'CDP1'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
  sends PLAY to exit pause
  sends STOP for pause
     CDP1_DEFEAT_FEEDBACK = OFF
     CDP1_PAUSE_TO_STOP = 6000 (10 min)
     Current: Version 2.0

B. *CDP2*

***SYS*TEM_CALL [DECK] 'CDP2'**
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
  sends PLAY to exit pause
     CDP2_DEFEAT_FEEDBACK = OFF
     CDP2_PAUSE_TO_STOP = 6000 (10 min)
     Current: Version 2.0

C. CDP3

SYSTEM_CALL [DECK] 'CDP3'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
  sends PLAY to exit pause
  will not repeatedly send stop (use for decks with STOP/EJECT functions)
     CDP3_DEFEAT_FEEDBACK = OFF
     CDP3_PAUSE_TO_STOP = 6000 (10 min)
     Current: Version 2.0

D. CDP4

SYSTEM_CALL [DECK] 'CDP4'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
  sends PAUSE to exit pause
     CDP4_DEFEAT_FEEDBACK = OFF
     CDP4_PAUSE_TO_STOP = 6000 (10 min)
     Current: Version 2.0

E. CDP5

SYSTEM_CALL [DECK] 'CDP5'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
  play & pause are same function
     CDP5_DEFEAT_FEEDBACK = OFF
     CDP5_PAUSE_TO_STOP = 6000 (10 min)
     Current: Version 2.0

F. CDP6

V. DAT calls----------

```
*****************************************************************
******* Instancing is required on these calls! ******************
*****************************************************************
```

A. *DAT1*

SYSTEM_CALL [DECK] 'DAT1'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)

sends PLAY to exit play pause
sends PAUSE to exit record pause
skips from PLAY,STOP,PAUSE
ffwd/rew's from STOP
searches fwd/rev from PLAY
   DAT1_REW_TO_STOP = 1800 (3 min)
   DAT1_SREV_TO_STOP = 12000 (20 min)
   DA13_PAUSE_TO_STOP = 6000 (10 min)
   DAT1_PULSE_DELAY = 3 (0.3 sec)
   DAT1_DEFEAT_FEEDBACK = OFF
   Current: Version 2.0

VI.DVD calls----------

****
******* Instancing is required on these calls! **********

A. DVD1

SYSTEM_CALL [DECK] 'DVD1'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
sends PLAY to exit pause
sends STOP twice to execute STOP function
skips from PLAY,PAUSE
searches fwd/rev from PLAY
DVD1_PAUSE_TO_STOP = 6000 (10 min)
DVD1_PULSE_DELAY = 7 (0.7 sec)
Current: Version 2.0

B. DVD2

SYSTEM_CALL [DECK] 'DVD2'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
sends PLAY to exit pause
sends STOP twice to execute STOP function
skips from PLAY,PAUSE
searches fwd/rev from PLAY
DVD2_PAUSE_TO_STOP = 6000 (10 min)
DVD2_PULSE_DELAY = 10 (10.0 sec)
Current: Version 2.0

VII. JOYSTICK calls----------

A. JOY1

SYSTEM_CALL 'JOY1' (DEV,L,L_MIN,L_CENTER,L_MAX,L_LEV,L_FLIP)
  scales AI8 input from 0 to 255 as a SEND_LEVEL
     LAG = 10
     Current: Version 2.0

B. JOY2

SYSTEM_CALL 'JOY2'
(DEV,H,H_MIN,H_CENTER,H_MAX,H_LEV,H_FLIP,V,V_MIN,V_CENTER,V_MAX,V_LEV,
V_FLIP)
  scales AI8 input from 0 to 255 as a SEND_LEVEL
     LAG = 10
     Current: Version 2.0

VIII. LASER DISC calls----------

****
******* Instancing is required on these calls! **********

A. LDP1

SYSTEM_CALL [DECK] 'LDP1'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
uses Sony LDP series 232 protocol
sends FAST PLAY for searches
sends SCAN for ffwd/rew
LDP1_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

B. LDP2

SYSTEM_CALL [DECK] 'LDP2'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
uses Pioneer LDV4200 family 232 protocol
sends MULTI-SPEED PLAY for searches
sends SCAN for ffwd/rew
sends PAUSE for stop
sends STILL for pause
LDP2_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

IX.PRODIGY calls----------

****

******* Instancing is required on Some calls! **********

A. PRO1

SYSTEM_CALL [PRODIGY] 'PRO1' (PANEL,PRODIGY)

prodigy programmer (MLCD or EL)

EL+ pages called PRO1.PGZ. Located in your TPDesign 1.23 Directory.

PRODIGY_BUSY

PRODIGY_PRESET

Current: Version 2.0

B. PRO2

SYSTEM_CALL [PRODIGY] 'PRO2' (MSP,PRODIGY,MAXZONES)

emulates PRO-SP8

5 preset mode with record

PRODIGY_BUSY

PRODIGY_PRESET

PRODIGY_PRESET_TIME = 20 (2 sec)

Current: Version 2.0

C. PRO3

SYSTEM_CALL [PRODIGY] 'PRO3' (MSP,PRODIGY,MAXZONES)

emulates PRO-SP8

5 preset mode without record

PRODIGY_BUSY

PRODIGY_PRESET

Current: Version 2.0

D. PRO4

SYSTEM_CALL [PRODIGY] 'PRO4' (MSP,PRODIGY)

emulates PRO-SP8

8 preset recall mode

PRODIGY_BUSY

PRODIGY_PRESET

Current: Version 2.0

E. PRO5

SYSTEM_CALL [PRODIGY] 'PRO5' (MSP,PRODIGY,MAXZONES)

emulates PRO-SP8 using AXD-MSP8 (decora series)

5 preset mode with record

PRODIGY_BUSY

PRODIGY_PRESET

PRODIGY_PRESET_TIME = 20 (2 sec)

Current: Version 2.0

F. PRO6

SYSTEM_CALL [PRODIGY] 'PRO6' (MSP,PRODIGY,MAXZONES)

emulates PRO-SP8 using AXD-MSP8 (decora series)

5 preset mode without record

PRODIGY_BUSY

PRODIGY_PRESET

X. MISCELLANEOUS calls----------

A. ALL OFF

SYSTEM_CALL 'ALL OFF' (DECK,FIRST)
turns all channels off for standard functions (play..rec)
does not affect feedback
Current: Version 2.0

B. FEEDBACK

SYSTEM_CALL 'FEEDBACK' (DECK,FUNCTION,FIRST)
sets the feedback channels for the desired function
Current: Version 2.0

C. FUNCTION

SYSTEM_CALL 'FUNCTION' (DECK,FUNCTION,FIRST)
pulses the function and sets the feedback accordingly
Current: Version 2.0

D. LDP1F

SYSTEM_CALL 'LDP1F' (DECK,FUNCTION,FIRST)

E. LDP2F

SYSTEM_CALL 'LDP2F' (DECK,FUNCTION,FIRST)

F. VCR6F

SYSTEM_CALL 'VCR6F' (DECK,FUNCTION,FIRST)

G. VCR8F

SYSTEM_CALL 'VCR8F' (DECK,FUNCTION,FIRST)

H. VCR10F

SYSTEM_CALL 'VCR10F' (DECK,FUNCTION,FIRST)
sends the string for the function and sets the feedback accordingly
uses in conjunction with LDP1, LDP2, VCR6, VCR8 AND VCR10
Current: Version 2.0

XI.SCREEN calls ----------

```
*************************************************************
******* Instancing is required on these calls! ***************
*************************************************************
```

Any of the below calls can be used as a "FUNCTION" call. To actuate a screen up, down or stop relay as a function, (i.e. from within a macro or define call) make the panel device 0 and the first non-zero button number found will actuate the function the non-zero was found in. For instance, to make a screen go up, use the following line of code under a push or in a DEFINE_CALL:

```
SYSTEM_CALL [<instance>] 'SCREENx' (0,1,0,0,RELAY,SCR_UP,SCR_DN,SCR_STOP,0)
```

Where the device RELAY is the device with the screen relays and the constants SCR_UP, SCR_DN and SCR_STOP are the up, down and stop relays on the relay card. Since panel is zero and the Up button parameter is 1, the screen will go up.

A. SCREEN1

SYSTEM_CALL [CARD] 'SCREEN1' (PANEL,UPB,DNB,STOPB,CARD,UPR,DNR,STOPR,FIRST)

standard screen control for timed or momentary relays
provide break before make logic
stop relay is discrete and optional (use 0 if there is no stop)
STOP cancels UP or DOWN
Optional LOCKOUT flag keeps DOWN from executing while UP is in progress and vice-versa.
First parameter control screen run time.
SCREEN1_LOCKOUT = 0
SCREEN1_BREAK_TIME = 5 (1/2 second)
SCREEN1_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

B. SCREEN2

SYSTEM_CALL [CARD] 'SCREEN2' (PANEL,UPB,DNB,STOPB,CARD,UPR,DNR,STOPR,FIRST)

standard screen control for timed or momentary relays
provide break before make logic
stop relay use up AND down relays at the same time (use 0 for stop relay)
STOP cancels UP or DOWN
Optional LOCKOUT flag keeps DOWN from executing while UP is in progress and vice-versa.
First parameter control screen run time.
SCREEN2_LOCKOUT = 0
SCREEN2_BREAK_TIME = 5 (1/2 second)
SCREEN2_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

C. SCREEN3

SYSTEM_CALL[CARD] 'SCREEN3' (PANEL,UPB,DNB,STOPB,CARD,UPR,DNR,STOPR,FIRST)

standard screen control for latching screens
provide break before make logic
stop relay is discrete and optional (use 0 if there is no stop)
STOP cancels UP or DOWN
SCREEN3_BREAK_TIME = 5 (1/2 second)

SCREEN3_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

D. SCREEN4

SYSTEM_CALL [CARD] 'SCREEN4' (PANEL,UPB,DNB,STOPB,CARD,UPR,DNR,STOPR,FIRST)
standard screen control for timed or momentary relays
stop relay is discrete and optional (use 0 if there is no stop)
STOP cancels UP or DOWN
First parameter control screen run time.
SCREEN4_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

E. SCREEN5

SYSTEM_CALL [CARD] 'SCREEN6' (PANEL,UPB,DNB,STOPB,CARD,UPR,DNR,STOPR,FIRST)
standard screen control for latching screens
stop relay is discrete and optional (use 0 if there is no stop)
STOP cancels UP or DOWN
SCREEN3_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

XII. SLIDE calls----------

SYSTEM_CALL 'SLD1' (CARD,PANEL,FWDB,REVB,FINB,FOUTB,PONB,POFFB,FIRST)

standard slide functions

SLD1_DEFEAT_FEEDBACK = OFF

Current: Version 2.0

SYSTEM_CALL 'SLD2' (CARD1,CARD2,CARD3,PANEL,FWDB,REVB,FIRST1,FIRST2,FIRST3)

multiple projector all fwd/all rev buttons

SLD2_DEFEAT_FEEDBACK = OFF

SLD2_DEFEAT_POWER = OFF

Current: Version 2.0

XIII. SWITCHER calls----------

A. SWT1

SYSTEM_CALL 'SWT1' (CARD,INPUT,OUTPUT,LEVEL)
uses AutoPatch X series protocol
Current: Version 2.0

B. SWT2

SYSTEM_CALL 'SWT2' (CARD,INPUT,OUTPUT,LEVEL)
uses AutoPatch 4Y series protocol
sends literal switch commands
Current: Version 2.0

C. SWT3

SYSTEM_CALL 'SWT3' (CARD,INPUT,OUTPUT,LEVEL)
uses AutoPatch 4Y series protocol
sends logical switch commands (TAKE mode)
Current: Version 2.0

D. SWT4

SYSTEM_CALL 'SWT4' (CARD,INPUT,OUTPUT,LEVEL)
uses Hedco SCE-101 series protocol
Current: Version 2.0

E. SWT5

SYSTEM_CALL 'SWT5' (CARD,INPUT,OUTPUT,LEVEL)
uses Utah Scientific series protocol
Current: Version 2.0

F. SWT6

SYSTEM_CALL 'SWT6' (CARD,INPUT,OUTPUT,LEVEL)
uses AutoPatch 1Y series protocol
Current: Version 2.0

G. SWT7

SYSTEM_CALL 'SWT7' (CARD,INPUT,OUTPUT,LEVEL)
uses Sigma SCI-210 series protocol
Current: Version 2.0

H. SWT8

SYSTEM_CALL 'SWT8' (CARD,INPUT,OUTPUT,LEVEL)
uses Sigma 2100 series protocol
Current: Version 2.0

I. SWT9

SYSTEM_CALL 'SWT9' (CARD,INPUT,OUTPUT,LEVEL)
uses Sigma 2100 SCI 8x8, 16x16 series protocol
Current: Version 2.0

J. SWT10

SYSTEM_CALL 'SWT10' (CARD,INPUT,OUTPUT,LEVEL)

uses Extron 4LD, System 8 and System 10 series protocol
Current: Version 2.0

K. SWT11

SYSTEM_CALL 'SWT11' (CARD,INPUT,OUTPUT,ADR)
uses Barco RCDVS05 series protocol
Do not use projector address 2
Current: Version 2.0

L. SWT12

SYSTEM_CALL 'SWT12' (CARD,INPUT,OUTPUT,LEVEL)
uses Sierra 8 and 16 series Host #1 protocol
Current: Version 2.0

M. ***SWT13***

SYSTEM_CALL 'SWT13' (CARD,INPUT,OUTPUT,ADR)
uses INLINE IN1222,IN1422,IN1510,IN1710 series protocol
controls up to 4 units
Current: Version 2.0

N. SWT14

SYSTEM_CALL 'SWT14' (CARD,INPUT,OUTPUT,ADR)
Uses NEC 6PG series protocol
Current: Version 2.0

O. SWT15

SYSTEM_CALL 'SWT15' (CARD,INPUT,OUTPUT,PLANE)
uses Extron 200 series protocol
Current: Version 2.0

P. SWT16

SYSTEM_CALL 'SWT16' (CARD,INPUT,OUTPUT,LEVEL,ADR)
uses Telect 2000 series protocol
Current: Version 2.0

XIV. VCR calls----------

****
******* Instancing is required on these calls! **********

A. VCR1

SYSTEM_CALL [DECK] 'VCR1'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
has discrete searches
sends PLAY (or REC) to exit pause
VCR1_REW_TO_STOP = 1800 (3 min)
VCR1_SREV_TO_STOP = 12000 (20 min)
VCR1_PAUSE_TO_STOP = 6000 (10 min)
VCR1_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

B. VCR2

SYSTEM_CALL [DECK] 'VCR2'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
does not have discrete searches
sends PLAY (or REC) to exit pause
VCR2_REW_TO_STOP = 1800 (3 min)
VCR2_SREV_TO_STOP = 12000 (20 min)
VCR2_PAUSE_TO_STOP = 6000 (10 min)
VCR2_PULSE_DELAY = 3 (0.3 sec)
VCR2_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

C. VCR3

SYSTEM_CALL [DECK] 'VCR3'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
does not have discrete searches
has latching searches
sends PAUSE to exit pause
VCR3_REW_TO_STOP = 1800 (3 min)
VCR3_SREV_TO_STOP = 12000 (20 min)
VCR3_PAUSE_TO_STOP = 6000 (10 min)
VCR3_PULSE_DELAY = 3 (0.3 sec)
VCR3_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

D. VCR4

SYSTEM_CALL [DECK] 'VCR4'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
does not have discrete searches
sends PAUSE to exit pause
VCR4_REW_TO_STOP = 1800 (3 min)
VCR4_SREV_TO_STOP = 12000 (20 min)
VCR4_PAUSE_TO_STOP = 6000 (10 min)
VCR4_PULSE_DELAY = 3 (0.3 sec)
VCR4_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

E. VCR5

SYSTEM_CALL [DECK] 'VCR5'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
has discrete searches
sends PAUSE to exit pause
VCR5_REW_TO_STOP = 1800 (3 min)
VCR5_SREV_TO_STOP = 12000 (20 min)
VCR5_PAUSE_TO_STOP = 6000 (10 min)
VCR5_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

F. VCR6

SYSTEM_CALL [DECK] 'VCR6'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
uses Sony U-matic SP series 422 protocol
VCR6_REW_TO_STOP = 1800 (3 min)
VCR6_SREV_TO_STOP = 12000 (20 min)
VCR6_PAUSE_TO_STOP = 6000 (10 min)
VCR6_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

G. VCR7

SYSTEM_CALL [DECK] 'VCR7'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
does not have discrete searches
has latching searches
sends PLAY (or REC) to exit pause
VCR7_REW_TO_STOP = 1800 (3 min)
VCR7_SREV_TO_STOP = 12000 (20 min)
VCR7_PAUSE_TO_STOP = 6000 (10 min)
VCR7_PULSE_DELAY = 3 (0.3 sec)
VCR7_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

H. VCR8

SYSTEM_CALL [DECK] 'VCR8'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
uses SONY BKU-701 232 protocol
VCR8_REW_TO_STOP = 1800 (3 min)
VCR8_SREV_TO_STOP = 12000 (20 min)
VCR8_PAUSE_TO_STOP = 6000 (10 min)
VCR8_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

I. VCR9

SYSTEM_CALL [DECK] 'VCR9'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)
has discrete searches
sends PLAY and REC to record
latches SFWD and SREV relays
sends PAUSE to exit PAUSE
uses PANASONIC relay controlled VCRs
VCR9_REW_TO_STOP = 1800 (3 min)
VCR9_SREV_TO_STOP = 12000 (20 min)

VCR9_PAUSE_TO_STOP = 6000 (10 min)
VCR9_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

J. VCR10

SYSTEM_CALL [DECK] 'VCR10'
(DECK,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,RECB,FIRST)

does not have discrete searches
uses VAR FWD at 0 speed for PAUSE
uses VAR FWD/REW for searches
uses STOP as a REC PAUSE
uses SONY UMATIC SP series 232 protocol
VCR10_REW_TO_STOP = 1800 (3 min)
VCR10_SREV_TO_STOP = 12000 (20 min)
VCR10_PAUSE_TO_STOP = 6000 (10 min)
VCR10_DEFEAT_FEEDBACK = OFF
Current: Version 2.0

XV. VOLUME calls----------

A. VOL1

SYSTEM_CALL 'VOL1' (PANEL,UPB,DNB,MUTEB,CARD,UPC,DNC,MUTEC)

```
  standard volume control
  mute is toggling
  UP or DOWN cancels mute
     VOL1_DEFEAT_FEEDBACK = OFF
     Current: Version 2.0
```

B. VOL2

SYSTEM_CALL 'VOL2' (PANEL,UPB,DNB,MUTEB,CARD,UPC,DNC)

```
  volume control for VX-1
  UP & DOWN latch for mute
  mute is toggling
  UP or DOWN cancels mute
     VOL2_DEFEAT_FEEDBACK = OFF
     Current: Version 2.0
```

XVI. Video Projector Calls----------

****
******* Instancing is required on these calls! **********

A. VPJ1

SYSTEM_CALL [DEV] 'VPJ1' (DEV,ADDR,FUNCTION,DATA,PANEL,BTNNUM)

NEC PG Video Projectors
Current: Version 2.0

B. VPJ2

SYSTEM_CALL [DEV] 'VPJ2' (DEV,ADDR,FUNCTION,DATA,PANEL,BTNNUM)

NEC PG+ or PG XTRA Video Projectors
Current: Version 2.0

C. VPJ6

SYSTEM_CALL [DEV] 'VPJ6' (DEV,ADDR,FUNCTION,DATA,PANEL,BTNNUM)

BARCO CRT Video Projectors
Current: Version 2.0

D. VPJ7

SYSTEM_CALL [DEV] 'VPJ7' (DEV,ADDR,FUNCTION,DATA,PANEL,BTNNUM)

BARCO LCD Video Projectors
Current: Version 2.0

E. VPJ11

SYSTEM_CALL [DEV] 'VPJ11' (DEV,ADDR,FUNCTION,DATA,PANEL,BTNNUM)

ELECTROHOME MARQUEE Video Projectors
Current: Version 2.0

```
Where :
      DEV         = AMX DEVICE NUMBER
      ADDR        = VIDEO PROJECTOR ADDRESS (1-8)
      FUNCTION    = VALUE OF FUNCTION
            VP_POWER_SET      = 1
            VP_VIDEO_MUTE_SET = 2
            VP_AUDIO_MUTE_SET = 3
            VP_INPUT_SELECT   = 4
            VP_CHANNEL_SELECT = 5
            VP_VOLUME_SET     = 10
            VP_BRIGHT_SET     = 11
            VP_COLOR_SET      = 12
            VP_CONTR_SET      = 13
            VP_SHARP_SET      = 14
            VP_TINT_SET       = 15
            VP_VOLUME_RAMP    = 20
            VP_BRIGHT_RAMP    = 21
            VP_COLOR_RAMP     = 22
            VP_CONTR_RAMP     = 23
            VP_SHARP_RAMP     = 24
            VP_TINT_RAMP      = 25
      DATA        = ASSOCIATED DATA
            VP_SET_ON     = 1
            VP_SET_OFF    = 2
```

```
      VP_SET_TOGGLE = 3
      VP_INCREASE   = 4
      VP_DECREASE   = 5
PANEL       = AMX CONTROL PANEL (OR 0)
BTNNUM      = PANEL BUTTON NUMBER (OR 0)
```

XVII. MSG call----------

SYSTEM_CALL 'MSG' (__FILE__,__DATE__,__TIME__,__NAME__,"compiler rev")

Where:

__FILE__ : DOS filename
__DATE__ : FILE DATE STAMP
__TIME__ : FILE TIME STAMP
__NAME__ : PROGRAM NAME
"compiler rev" : String representing compiler rev #, e.g. '3.04'
If using AXCESS 3.05 or greater, "compiler rev" can be substituted with __VERSION__. __VERSION__ automatically provides the compiler rev.

Current: Version 2.0

XVIII. LONGDATE call----------

SYSTEM_CALL 'LONGDATE' (DATE_STR)

```
Where:
      DATE_STR is an array variable with length 10 or more

Why:
      This call returns a date formatted string similar to
      DATE but the year is 4 digits. This call will return
      the 4 digit year correctly up through 2079. An example
      string from LONGDATE is:  07/21/2017

Example:

DEFINE_VARIBLE
DATE_STR[12]                    (* Long date from LONGDATE.lib *)
.
.
.
DEFINE_PROGRAM

SYSTEM_CALL 'LONGDATE' (DATE_STR)  (* DATE_STR now contains date with *)
                           (* month/day/4 digit year *)

Note: WILDCARDS (i.e '08-05-??') will not work with the string
      returned from LONGDATE

Current: Version 2.0
```

XIX. HEXTOI call----------

SYSTEM_CALL 'HEXTOI' (STRING,VALUE)

```
Where:
      STRING is an array variable with length 4 or more
       containing and ASCII-HEX string (for instance, 'FFFF')
      VALUE is a return integer containing the value of
       the STRING when interpreted as HEX

Example:

DEFINE_VARIBLE
HEX_STR[4]                (* ASCII-HEX string *)
VALUE                     (* Value of Above *)
.
.
.
DEFINE_PROGRAM

HEX_STR = '00FF'
SYSTEM_CALL 'HEXTOI' (HEX_STRING,VALUE)
IF (VALUE = $FF)          (* This is True! *)
{
  SEND_STRING 0,"'Value is = $FF (255),',13,10"
}

Current: Version 2.01
```

XX. AMPMTIME call----------

SYSTEM_CALL 'AMPMTIME' (STRING)

```
Where:
      STRING is an array variable with length 5 or more.
       AM/PM Formatted time will be returned.

Example:

DEFINE_VARIBLE
TIME_STR[8]               (* Time string *)
.
.
.
DEFINE_PROGRAM

SYSTEM_CALL 'AMPMTIME' (TIME_STR)
(* TIME_STR will now be a time format like: '5:14 PM' *)

Current: Version 2.00
```

XXI. DAYLTSAV call----------

SYSTEM_CALL 'DAYLTSAV'

This call performs daylight savings time changes. The system time will be changed according to the following rules:

THE FOLLOWING EXCERPT IS OFFERED AS AN EXPLANATION AND IS FROM:
http://www4.law.cornell.edu/uscode/15/260a.html
United States Code
TITLE 15 - COMMERCE AND TRADE
CHAPTER 6 - WEIGHTS AND MEASURES AND STANDARD TIME
SUBCHAPTER IX - STANDARD TIME
Sec. 260a. Advancement of time or changeover dates

(a) Duration of period; State exemption
During the period commencing at 2 o'clock antemeridian on the first Sunday of April of each year and ending at 2 o'clock antemeridian on the last Sunday of October of each year, the standard time of each zone established by sections 261 to 264 of this of this title, as modified by section 265 of this title, shall be advanced one hour and such time as so advanced shall for the purposes of such sections 261 to 264, as so modified, be the standard time of such zone during such period; however, (1) any State that lies entirely within one time zone may by law exempt itself from the provisions of this subsection providing for the advancement of time, but only if that law provides that the entire State (including all political subdivisions thereof) shall observe the standard time otherwise applicable during that period, and (2) any State with parts thereof in more than one time zone may by law exempt either the entire State as provided in (1) or may exempt the entire area of the State lying within any time zone.

Example:

```
DEFINE_PROGRAM

SYSTEM_CALL 'DAYLTSAV'
```

Current: Version 2.00

XXII. NEW FORMAT CALLS -

A. Pioneer CLDV Series - Version 2.00 ----------

****
******* Instancing is required on these calls! **********

1. PIOLD000

SYSTEM_CALL [LDP] 'PIOLD000' (LDP)
Initialization call. This call should always be included
at start up. It initializes the laser disc and the system call.

2. PIOLD001

SYSTEM_CALL [LDP] 'PIOLD001'
(LDP,PANEL,PLAYB,STOPB,PAUSEB,FFWDB,REWB,SFWDB,SREVB,FIRST)
Basic transport control. Parameters are:
Play, Stop, Pause, Chp Fwd, Chp Rev, Scan Fwd, Scan Rev, First

3. PIOLD002

SYSTEM_CALL [LDP] 'PIOLD002'
(LDP,PANEL,B0,B9,CHPB,FRMB,CLRB,SEB,VT_ADDR,VT_MODE,VT_CHP,VT_FRM)
Search routine. Parameters are:
B0 and B9 define are the 0 button and the 9 button on a ten-key keypad.
Number 0-9 must be ascending and contiguous.
CHPB is the Chapter or Track mode select button
FRMB is the Frame or Time mode select button
CLRB is the clear button
SEB is the search button
VT_ADDR is the variable text channel for the search address display
VT_MODE is the variable text channel for the search mode display
VT_CHP is the variable text channel for the CHPB button*
VT_FRM is the variable text channel for the FRMB button*
* This call changes the function of the CHPB and FRB to allow only the
search mode for the particular disc type. Variable text codes are needed
to display the name of the function that the button achieves.
See the BNAME constants below to change the text these buttons receive.

4. PIOLD00B

SYSTEM_CALL [LDP] 'PIOLD00B' (LDP,LDP_BUFFER)
Buffer processing. This call should always be included
in mainline. It communicates with the laser disc player.
The LDP_BUFFER is a variable that must be created
in your DEFINE_VARIABLE section and you must assign it to
be a buffer in DEFINE_START with a line like:

CREATE_BUFFER LDP,LDP_BUFFER

5. PIOLD00F

SYSTEM_CALL [LDP] 'PIOLD00F' (LDP,PARAMETER)
Laser Disc function call. Any of the following can be passed
as PARAMETER to achieve a desired laser disc function.

```
     PIOLD000_FFWD_STOP        = 260    (* 4 | $100 *)
     PIOLD000_REW_STOP         = 261    (* 5 | $100 *)
```

```
      PIOLD000_SFWD_STOP        = 262    (* 6 | $100 *)
      PIOLD000_SREV_STOP        = 263    (* 7 | $100 *)
      PIOLD000_STILL_ST_FWD     = 8
      PIOLD000_STILL_ST_REV     = 9
      PIOLD000_MULTI_SP_FWD     = 10
      PIOLD000_MULTI_SP_REV     = 11
      PIOLD000_STILL_ST_FSTOP   = 264    (* 8 | $100 *)
      PIOLD000_STILL_ST_RSTOP   = 265    (* 9 | $100 *)
      PIOLD000_STILL_ST_FWD_FB  = 236
      PIOLD000_STILL_ST_REV_FB  = 237
      PIOLD000_MULTI_SP_FWD_FB  = 238
      PIOLD000_MULTI_SP_REV_FB  = 239
      PIOLD000_DISPLAY_ON       = 61
      PIOLD000_DISPLAY_OFF      = 62
      PIOLD000_DOOR_OPEN        = 63     (* 1 IF DOOR IS OPEN *)
      PIOLD000_DOOR_CLOSE       = 64     (* 1 IF DOOR IS CLOSED *)
      PIOLD000_VIDEO_MUTE       = 65     (* 1 IF VIDEO IS MUTED *)
      PIOLD000_VIDEO_UNMUTE     = 66     (* 1 IF VIDEO IS UNMUTED *)

The following constants can be overridden:
Button names:
PIOLD000_CHAPTER_BNAME    = 'CHAPTER'
PIOLD000_FRAME_BNAME      = 'FRAME'
PIOLD000_TIME_BNAME       = 'TIME'
PIOLD000_TRACK_BNAME      = 'TRACK'
PIOLD000_CHP_TIME_BNAME   = 'CHAPTER/TIME'
PIOLD000_CHP_FRAME_BNAME  = 'CHAPTER/FRAME'
PIOLD000_TRACK_TIME_BNAME = 'TRACK/TIME'
PIOLD000_NONE_BNAME       = 'NONE'
  These hold the text that will sent to the CHPB and FRMB buttons from the
PIOLD002 call.
  You can redefine these in the constant section to change the text on these
buttons
  like this:

  DEFINE_CONSTANT
  PIOLD000_CHAPTER_BNAME = 'New Chapter Text'

PIOLD000_MAX_OCC          = 10
  This call supports 10 laser disc players by default.  Redefine this constant
  to increase this number.

PIOLD002_FLASH_TIME       = 5
  Adjusts the rate that the Search button flashes during a search.

PIOLD002_SE_ERROR_TIME    = 20
  Adjusts the time that the message ERROR is displayed on a search error.

PIOLD000_STEP_REP_TIME    = 5
  Adjust the rate of the step repeats.

PIOLD000_SCAN_REP_TIME    = 3
  Adjust the rate of the scan repeats.

PIOLD001_DEFEAT_FEEDBACK = 0
PIOLD002_DEFEAT_FEEDBACK = 0
  Defeats feedback for each of the calls listed when the constant
```

```
  is defined as non-zero (usually 1).

STATUS
  Status can read from the following channels if needed:
      PIOLD000_CLV              = 76
      PIOLD000_CAV              = 77
      PIOLD000_CD               = 78
      PIOLD000_CDV              = 79
      PIOLD000_MOTOR_ON         = 80
      PIOLD000_DISC_LOADED      = 81
      PIOLD000_HAS_CHP          = 82
      PIOLD000_SIDE1            = 83
      PIOLD000_SIDE2            = 84
      PIOLD000_8IN              = 85
      PIOLD000_12IN             = 86
      PIOLD000_SEARCH_PEND      = 87
      PIOLD000_RECEIVED_ACK     = 88
```

6. EXAMPLE PROGRAM:

```
(*******************************************************)
(*          DEVICE NUMBER DEFINITIONS GO BELOW         *)
(*******************************************************)
DEFINE_DEVICE

LDP                 = 1           (* AXC-232: PIONEER CLDV SERIES *)
TP                  = 128         (* AXT-PANEL *)

(*******************************************************)
(*               CONSTANT DEFINITIONS GO BELOW         *)
(*******************************************************)
DEFINE_CONSTANT

(*******************************************************)
(*               VARIABLE DEFINITIONS GO BELOW         *)
(*******************************************************)
DEFINE_VARIABLE

(* LDP *)
LDP_BUFFER[100]                   (* INCOMING BUFFER *)

(*******************************************************)
(*               LATCHING DEFINITIONS GO BELOW         *)
(*******************************************************)
DEFINE_LATCHING

(*******************************************************)
(*       MUTUALLY EXCLUSIVE DEFINITIONS GO BELOW       *)
(*******************************************************)
DEFINE_MUTUALLY_EXCLUSIVE

(*******************************************************)
(*           SUBROUTINE DEFINITIONS GO BELOW           *)
(*******************************************************)

(*******************************************************)
```

```
(*                STARTUP CODE GOES BELOW                    *)
(***********************************************************)
DEFINE_START

(* LDP *)
CREATE_BUFFER LDP,LDP_BUFFER
SYSTEM_CALL [LDP] 'PIOLD000' (LDP)

(***********************************************************)
(*            THE ACTUAL PROGRAM GOES BELOW                *)
(***********************************************************)
DEFINE_PROGRAM

(* LASER DISC TRANSPORT FUNCTIONS *************************)
SYSTEM_CALL [LDP] 'PIOLD001' (LDP,TP,151,152,153,154,155,188,189,0)
SYSTEM_CALL [LDP] 'PIOLD002' (LDP,TP,10,19,20,21,22,23,1,2,3,4)
SYSTEM_CALL [LDP] 'PIOLD00B' (LDP,LDP_BUFFER)

(***********************************************************)
(*                     END OF PROGRAM                      *)
(*        DO NOT PUT ANY CODE BELOW THIS COMMENT           *)
(***********************************************************)
```

B. Peavy SYSTEM_CALLS

Peavey Mediamatrix - Version 2.00 ----------

NOTE: - Peavey user id's (UID) are required for each level to be controlled!
- -Correct values are (ascii):
  - 000 through FFF - Always 3 digits within this range!
- These UID's are stored within the pasha.ini file on the Mediamatrix computer!

- System calls support two different types of levels.
- First, a single user id is supported where a level of 0 is sent for mute.
- Second, a dual user id is supported where there is a uid for level and another uid for mute for the same channel of volume. Use PVYVO00R to register the mute id!!

1. PVYVO001

SYSTEM_CALL 'PVYVO001' (CARD,PANEL,UPB,DNB,MUTEB,P1B,P2B,P3B,P4B,SPB,UID[3])

Basic volume up/down/mute/preset control. Button parameters are:
UP, DOWN, MUTE, PRESET 1, PRESET 2, PRESET 3, PRESET 4, SAVE PRESET

2. PVYVO00L

SYSTEM_CALL 'PVYVO00L' (CARD,UID[3],LVL)

(level) Use this call in mainline to passback the system_call volume level to the caller (for bargraph display purposes).
CARD - AMX device connected to Peavey
UID - User ID of the level to display
LVL - Passed back volume level to caller

3. PVYVO00P

SYSTEM_CALL 'PVYVO00P' (CARD,UID[3],LVL)

```
  (Preset) Use this call when user needs to define more than the 4 presets
  that the system call will keep track of.
  CARD - AMX device connected to Peavey
  UID  - User ID of the level to send to preset
  LVL  - New volume level
```

4. PVYVO00F

```
SYSTEM_CALL 'PVYVO00F' (CARD,UID[3],FN)
  (Function) Used to mute, unmute, goto preset 1,2,3,4 for a chosen level.
  Use these functions:
  PVYVO000_MUTEC   =  3            (* (Mute on=3), (Mute off=3|$100) *)
  PVYVO000_PSET1   =  11
  PVYVO000_PSET2   =  12
  PVYVO000_PSET3   =  13
  PVYVO000_PSET4   =  14
```

5. PVYVO00R

```
SYSTEM_CALL 'PVYVO00R' (CARD,OCC,L_UID[3],M_UID[3],LVL_PTR)
  (Register) Used in STARTUP to register a separate mute user id for a given
  level.  After the level is registered, system call will lookup the value!
  CARD     - AMX device connected to Peavey
  OCC      - Use 0 (Advanced feature)
  L_UID    - User ID of the level to send to preset
  M_UID    - User ID of the mute that is to be registered
  LVL_PTR  - Use 0 (Advanced feature)
```

6. EXAMPLE CODE GOES BELOW

```
PROGRAM_NAME='PVYVO000 - SYSTEM CALL EXAMPLE (TYPICAL)'
(*   DATE:08/17/98    TIME:10:10:03    *)
(***********************************************************)
(*          DEVICE NUMBER DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_DEVICE

PEAVEY     =  1     (* AXC-232 MEDIAMATRIX *)
TP         =  128   (* SOME PANEL *)

(***********************************************************)
(*               CONSTANT DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_CONSTANT

#IF_NOT_DEFINED PVYVO000_VOL_FUNCTION
PVYVO000_UP      =  1        (* FB FLAGS *)
PVYVO000_DN      =  2
PVYVO000_MUTE    =  3

PVYVO000_PSET1   =  11       (* PRESET FLAGS *)
PVYVO000_PSET2   =  12
PVYVO000_PSET3   =  13
PVYVO000_PSET4   =  14
#END_IF
```

```
(***********************************************************)
(*               VARIABLE DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_VARIABLE

PGM_LVL
MIC_LVL

(***********************************************************)
(*               LATCHING DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_LATCHING

(***********************************************************)
(*       MUTUALLY EXCLUSIVE DEFINITIONS GO BELOW           *)
(***********************************************************)
DEFINE_MUTUALLY_EXCLUSIVE

(***********************************************************)
(*           SUBROUTINE DEFINITIONS GO BELOW               *)
(***********************************************************)


(***********************************************************)
(*                STARTUP CODE GOES BELOW                  *)
(***********************************************************)
DEFINE_START

(* NOTE: This example will use different User ID's (UID) for level #1.
         Level #1 ramping will use UID 001.  Level #1 mute will use UID 111.
         In order to do this within the system call, the mute UID must be
         registered in Startup like below.  Parameters that are set to 0
         in the PVYVO00R call can be ignored.  They are used with advanced
         configurations!                                                  *)
SYSTEM_CALL 'PVYVO00R' (PEAVEY,0,'001','111',0)


(***********************************************************)
(*            THE ACTUAL PROGRAM GOES BELOW                *)
(***********************************************************)
DEFINE_PROGRAM

(***** LEVEL #1 ('001') ***************************************************)
(* VOLUME UP/DN/MUTE/PSET *)
(* NOTE: PSET STORE BASED UPON STORE BUTTON (8).. *)
SYSTEM_CALL 'PVYVO001' (PEAVEY,TP,1,2,3,4,5,6,7,8,'001')

(* GET VOLUME LEVEL FROM SYSTEM_CALL TO DISPLAY *)
SYSTEM_CALL 'PVYVO00L'    (PEAVEY,'001',PGM_LVL)
SEND_LEVEL TP,1,PGM_LVL


(***** LEVEL #2 ('FFF') ***************************************************)
(* VOLUME UP/DN/MUTE/PSET *)
(* NOTE: PSET STORE BASED UPON PRESS AND HOLD TO STORE.. *)
SYSTEM_CALL 'PVYVO001' (PEAVEY,TP,9,10,11,13,14,15,16,0,'FFF')
```

```
(* GET VOLUME LEVEL FROM SYSTEM_CALL TO DISPLAY *)
SYSTEM_CALL 'PVYVO00L' (PEAVEY,'FFF',MIC_LVL)
SEND_LEVEL TP,2,MIC_LVL


(* SET NEW VOLUME LEVEL FOR LEVEL #1 (USER DEFINED PRESETS) *****************)
PUSH[TP,63]
{
  PGM_LVL = 64
  SYSTEM_CALL 'PVYVO00P' (PEAVEY,'001',PGM_LVL)
}


(* SET NEW VOLUME LEVEL FOR LEVEL #2 (USER DEFINED PRESETS) *****************)
PUSH[TP,67]
{
  PGM_LVL = 255
  SYSTEM_CALL 'PVYVO00P' (PEAVEY,'FFF',PGM_LVL)
}


(* FUNCTION CALL TO RECALL PSETS 1-4 FOR LEVEL #1 (SYS CALL DEFINED) ********)
PUSH[TP,71]
  SYSTEM_CALL 'PVYVO00F' (PEAVEY,'001',PVYVO000_PSET1)
PUSH[TP,72]
  SYSTEM_CALL 'PVYVO00F' (PEAVEY,'001',PVYVO000_PSET2)
PUSH[TP,73]
  SYSTEM_CALL 'PVYVO00F' (PEAVEY,'001',PVYVO000_PSET3)
PUSH[TP,74]
  SYSTEM_CALL 'PVYVO00F' (PEAVEY,'001',PVYVO000_PSET4)


(* FUNCTION CALL TO RECALL PSETS 1-4 FOR LEVEL #2 (SYS CALL DEFINED) ********)
PUSH[TP,81]

  SYSTEM_CALL 'PVYVO00F' (PEAVEY,'FFF',PVYVO000_PSET1)
PUSH[TP,82]
  SYSTEM_CALL 'PVYVO00F' (PEAVEY,'FFF',PVYVO000_PSET2)
PUSH[TP,83]
  SYSTEM_CALL 'PVYVO00F' (PEAVEY,'FFF',PVYVO000_PSET3)
PUSH[TP,84]
  SYSTEM_CALL 'PVYVO00F' (PEAVEY,'FFF',PVYVO000_PSET4)

(*********************************************************)
(*                     END OF PROGRAM                    *)
(*        DO NOT PUT ANY CODE BELOW THIS COMMENT         *)
(*********************************************************)
```

XXIII. OLD CALLS ----------

CORRESPONDENCE OF OLD SYSTEM CALLS TO NEW SYSTEM CALLS:
*(Note: Some of these system calls are rewrites of older system calls and others are a "closest fit".) New System Calls, with a few exceptions, shall use the following naming convention: three letters to indicate the type of equipment and one or two numbers to distinguish calls within a type of equipment.

"OLDER"	"NEWER"
AUTOPATCH X:	USE SWT1
AUTOPATCH Y:	USE SWT2 OR SWT3
CASS_5:	USE CAS1
CASS2_5:	USE CAS2
CASS3_5:	USE CAS3
CASS3_5R:	USE CAS2
CASS4_5R:	USE CAS3
CD1_5:	USE CDP1
CD2_5:	USE CDP2
CD3_5:	USE CDP1
CD4_4:	OBSOLETE SYSTEM CALL
CD5_5:	USE CDP5
CD6_5:	USE CDP2
DO_MACRO:	OBSOLETE SYSTEM CALL
FAR END CONTROL:	OBSOLETE SYSTEM CALL
OLD FEEDBACK:	USE FEEDBACK
HEDCO16X:	USE SWT4
INIT DECK:	USE FUNCTION
IR_LDV_5:	USE CDP1
SONYLD_7:	USE LDP1
PIONEER_LDV_5:	USE LDP2
PIONEER_LDV_7:	USE LDP2
PIONEER_LDP1_5:	OBSOLETE SYSTEM CALL
PNRLD1_7:	OBSOLETE SYSTEM CALL
PRODIGY PROGRAMMER:	USE PRO1
TYPE5_5:	USE VCR5
TYPE5_5R:	USE VCR5
TYPE5_7:	USE VCR5
TYPE6_5:	USE VCR6
TYPE6_7R:	USE VCR6
TYPE7_5:	OBSOLETE SYSTEM CALL
TYPE8_5:	USE VCR8
TYPE8_7:	USE VCR8
DO TYPEWRITER:	OBSOLETE SYSTEM CALL
UTAH SCI:	USE SWT5
VOLUME PRESET:	OBSOLETE SYSTEM CALL
VOLUME_2:	USE VOL1
SLIDES:	USE SLD1
SLIDES_2:	USE SLD1
SLIDES_3:	USE SLD1
SONY8_5:	USE VCR8
TYPE1_5:	USE VCR1

```
TYPE1_5R:                       USE VCR1
TYPE1_7:                        USE VCR1
TYPE1_7R:                       USE VCR1
TYPE2_5:                        USE VCR2
TYPE2_5R:                       USE VCR2
TYPE2_7:                        USE VCR2
TYPE3_5:                        USE VCR3
TYPE3_5R:                       USE VCR3
TYPE3_7:                        USE VCR3
TYPE4_5:                        USE VCR4
TYPE4_5R:                       USE VCR4
```

5. Button Parameters -

For all of the button parameters (parameters ending with a B, such as PLAYB) simply pass the button number of the panel that should activate that function. If a particular button does not exist, pass a 0 in its place. Certain action may be assumed when this happens. To override this feature, pass a 256 instead of a 0 for a function to disable it from occurring under another button push. For example, to control Auto Focus with only 1 button, pass the button code for the AFB (auto focus button) and pass a 0 for the MFB (manual focus button). The AFB will then have toggling action. To make the AFB only do auto focus, pass the button number for AFB and pass a 256 for the MFB. The AFB will only do Auto Focus.

6. Speeds -

The camera calls may define the constant MMMCA??0_PAN_SPEED, MMMCA??0_TILT_SPEED, MMMCA??0_ZOOM_SPEED, and MMMCA??0_FOCUS_SPEED. These constants can be overridden to change the default speed. A camera call might not define these if they are not adjustable. For example, the Sony EVI-D30 call (SONCA00X) defines the following constants: SONCA000_PAN_SPEED, SONCA000_TILT_SPEED, and SONCA000_ZOOM_SPEED. SONCA000_FOCUS_SPEED is not defined since the focus speed is not adjustable.

To override the speeds, define
the following constants in the DEFINE_CONSTANT section of your program:

```
DEFINE_CONSTANT
SONCA000_PAN_SPEED = 3
```

To make the speeds variable, define
the following variables in the DEFINE_VARIABLE section of your program and assign a value in DEFINE_START:

```
DEFINE_VARIABLE
SONCA000_PAN_SPEED

DEFINE_START
SONCA000_PAN_SPEED = 3
```

If you change the speed during in your program, the next command start will run at the speed you have set.

See the documentation for each call to see which speeds are adjustable and what the valid ranges are.

7. Camera Address -

Camera with adjustable address assume an address of 1 (or it's equivalent). This address is can be overridden by re-defining the constant. For instance, to make the Parkervision Camerman call

(PARCA000) use address $08 (dipswitch 1) instead of address $04 (dipswitch 0), define the following constant in your program:

```
DEFINE_CONSTANT
PARCA000_CAM_ADDR = $08
```

See the documentation for each call to see if address are supported and what the valid ranges are. However, the calls limit one camera address per card.

8. Custom Features -

Custom camera feature may be accessible. There are a series of constants defined for each camera call that implement custom camera features. For instance, the Canon VCC3 Has video/audio mute features. One of the is defined as:

```
CANCA000_VIDEO_MUTE_ON = 73
```

The active this feature of the camera, pass the custom camera parameter to the function call (in this case, CANCA00F.LIB). An example of this would be:

```
DEFINE_CONSTANT
CANCA000_VIDEO_MUTE_ON = 73

DEFINE_PROGRAM
PUSH[TP,1]                          (* CAMERA VIDEO MUTE ON *)
  SYSTEM_CALL [CAM] 'CANCA00F' (CAM,CANCA000_VIDEO_MUTE_ON)
```

As of 4/10/98, system calls are now Time-Stamped to indicate revision number. Using MS-DOS or Windows Explorer, you check the revision number of the system call by using the date and time. The date will reflect the date the call was released and the time will reflect the revision. For example, if the file's time was 2:01 AM, the it is revision 2.01. All calls below are noted with current revision available.

XXV. Sony EVI-D30 calls - Version 2.02 ----------

```
*****************************************************************
*******   Instancing is required on this class of calls! **********
*****************************************************************
```

A. I. SYSTEM_CALL [CAM] 'SONCA000' (CAM)

```
Initialization call.  This call should always be included
at start up.  It initializes the camera and the system call.
```

B. II. SYSTEM_CALL [CAM] 'SONCA001' (CAM,PANEL,PLR,PRB,TUB,TDB,ZTB,ZWB,FNB,FFB,AFB,MFB)

```
Basic camera control.  Parameters are:
Pan Left, Pan Right, Tilt Up, Tilt Down, Zoom Tele, Zoom Wide,
Focus Near, Focus Far, Auto Focus, Manual Focus.
```

C. III. SYSTEM_CALL [CAM] 'SONCA002' (CAM,PANEL,P1B,P2B,P3B,P4B,P5B,P6B,P7B,P8B,P9B,P10B,SPB)

```
Basic camera presets.  Parameters are:
Presets 1-10 and Save.  If 0 is passed for save button, preset
buttons will store presets when held for 2 seconds.  Preset buttons
will recall only if 256 is passed for SPB.
```

D. IV. SYSTEM_CALL [CAM] 'SONCA003' (CAM,PANEL,IOB,ICB,AIB,MIB,AGBONB,AGCOFFB)

```
Basic iris control. Parameters are:
Iris Open, Iris Close, Auto Iris, Manual Iris, AGC on, AGC off.
AGC on and AGC off are not implemented.
```

E. V. SYSTEM_CALL [CAM] 'SONCA004' (CAM,PANEL,SOB,SCB,ASB,MSB)

```
Basic shutter control. Parameters are:
Shutter Open, Shutter Close, Auto Shutter, Manual Shutter.
```

F. VI. SYSTEM_CALL [CAM] 'SONCA00B' (CAM,CAM_BUFFER)

```
Buffer processing.  This call should always be included
in mainline.  It communicates with the camera.
The CAMERA_BUFFER is a variable that must be created
in your DEFINE_VARIABLE section and you must assign it to
be a buffer in DEFINE_START with a line like:
CREATE_BUFFER CAM,CAM_BUFFER
```

G. VII. SYSTEM_CALL [CAM] 'SONCA00F' (CAM,PARAMETER)

```
Camera function call.  Any of the following cam be passed
as PARAMETER to achieve a desired camera function.
       PAN_RIGHT               = 31
       TILT_DN                 = 32
       PAN_LEFT                = 35
```

```
TILT_UP                 = 36
ZOOM_TELE               = 3
FOCUS_NEAR              = 4
ZOOM_WIDE               = 7
FOCUS_FAR               = 8
PAN_STOP                = 287          (* PAN RIGHT | $100 *)
TILT_STOP               = 288          (* TILT DN | $100 *)
ZOOM_STOP               = 259          (* ZOOM TELE | $100 *)
FOCUS_STOP              = 260          (* FOCUS NEAR | $100 *)
AUTO_FOCUS              = 10
MAN_FOCUS               = 11
HOME_POSITION           = 99
PRESET1                 = 101
PRESET2                 = 102
PRESET3                 = 103
PRESET4                 = 104
PRESET5                 = 105
PRESET6                 = 106
PRESET7                 = 107
PRESET8                 = 108
PRESET9                 = 109
PRESET10                = 110
SAVE PRESET1            = 357
SAVE PRESET2            = 358
SAVE PRESET3            = 359
SAVE PRESET4            = 360
SAVE PRESET5            = 361
SAVE PRESET6            = 362
SAVE PRESET7            = 363
SAVE PRESET8            = 364
SAVE PRESET9            = 365
SAVE PRESET10           = 366
IRIS_OPEN               = 1
IRIS_CLOSE              = 2
AUTO_IRIS               = 61
MAN_IRIS                = 62
IRIS_STOP               = 257          (* IRIS OPEN | $100 *)
SHUTTER_OPEN            = 65
SHUTTER_CLOSE           = 66
AUTO_SHUTTER            = 67
MAN_SHUTTER             = 68
SHUTTER_STOP            = 270          (* SHUTTER OPEN | $100 *)
POWER                   = 9
POWER_ON                = 27
POWER_OFF               = 28
SONCA000_RESET          = 88
SONCA000_INIT_NETWORK   = 89
```

H. VIII. SYSTEM_CALL [CAM] 'SONCA00P' (CAM,STATE,SENS_DEV,SENS_CHAN)

Basic power control. Parameters are:
Desired state (0=off, 1=on, 2=toggle), Sensing Device (device which VSS or PCS is attached), Sensing Channel (channel of SENS_DEV which VSS or PCS is connected). This camera has discrete power control so the power sensing is optional. Pass a 0 for both SENS_DEV

```
 and SENS_CHAN under normal conditions.

Additional presets are available by passing the following
parameters to the function call:
        SONCA000_PRESET_BEG             = 111       (* PRESETS 11 *)
           ... THRU ...
        SONCA000_PRESET_END             = 130       (* MAX 30 PRESETS *)

        SONCA000_SAVE_PRESET_BEG        = 367       (* PRESETS 11 *)
           ... THRU ...
        SONCA000_SAVE_PRESET_END        = 386       (* MAX 30 PRESETS *)

However, the maximum number of presets is determined by the constant:
        SONCA000_MAX_PRESETS             = 30
which can be overridden to any value between 1 and 255.  Do not us
0 and be aware that setting it to a value of less than 10 will cause
buttons 7, 8, 9, or 10 in system call SONCA002 not to work.

The following constants can be overridden:
SONCA000_PSET_HOLD_TIME = 20 (* 2 SECONDS *)
  Defines how long the preset button is held before a preset
  is stored when a 0 had been passed for the SPB for call SONCA002.

SONCA001_DEFEAT_FEEDBACK = 0
SONCA002_DEFEAT_FEEDBACK = 0
SONCA003_DEFEAT_FEEDBACK = 0
SONCA004_DEFEAT_FEEDBACK = 0
  Defeats feedback for each of the calls listed when the constant
  is defined as non-zero (usually 1)

SONCA000_CAM_ADDR        = 1
  Re-defines the default address.  However, when the Visca
  network is initialize, address is assigned automatically.
  Address 1 will always be the camera closest to the AMX system.

SONCA000_PAN_SPEED        = $0D             (* 1-$18, $18 IS MAX *)
SONCA000_TILT_SPEED       = $0A             (* 1-$14, $14 IS MAX *)
SONCA000_ZOOM_SPEED       = $05             (* 1-8, 8 IS MAX *)
  Re-defines the default speeds for the camera operations
  when the button passed to SONCA001 is pushed.
```

I. EXAMPLE PROGRAM:

```
(***********************************************************)
(*               DEVICE DEFINITIONS GO BELOW               *)
(***********************************************************)
DEFINE_DEVICE

CAMERA              = 1           (* SONY EVI-D30 *)
TP                  = 128         (* ANY TP *)
```

```
(***********************************************************)
(*               CONSTANT DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_CONSTANT

SONCA000_MAX_PRESETS      = 30

(***********************************************************)
(*             VARIABLE DEFINITIONS GO BELOW               *)
(***********************************************************)
DEFINE_VARIABLE

CAMERA_BUFF[255]                  (* INCOMING BUFFER *)
CAM_PRESET                        (* CURRENT CAMERA PRESET *)
PRESET_SAVE                       (* 1 TO SAVE *)
PRESET_FUNC                       (* PRESET FUNCTION NUMBER *)

(***********************************************************)
(*             SUBROUTINE DEFINITIONS GO BELOW             *)
(***********************************************************)

(***********************************************************)
(*             START UP DEFINITIONS GO BELOW               *)
(***********************************************************)
DEFINE_START

CREATE_BUFFER CAMERA,CAMERA_BUFF
SYSTEM_CALL [CAMERA] 'SONCA000' (CAMERA)

(***********************************************************)
(*             DEFINE PROGRAM GOES BELOW                   *)
(***********************************************************)
DEFINE_PROGRAM

(*************************************)
(* PRESET METHOD #1 - 1 TO 10 PRESETS  *)
(*************************************)
SYSTEM_CALL [CAMERA] 'SONCA001' (CAMERA,TP,4,3,1,2,5,6,7,8,9,10)
SYSTEM_CALL [CAMERA] 'SONCA002' (CAMERA,TP,11,12,13,14,15,16,0,0,0,0,0)
SYSTEM_CALL [CAMERA] 'SONCA003' (CAMERA,TP,22,23,24,25,0,0)
SYSTEM_CALL [CAMERA] 'SONCA004' (CAMERA,TP,29,30,32,31)
SYSTEM_CALL [CAMERA] 'SONCA00B' (CAMERA,CAMERA_BUFF)

PUSH[TP,26]                       (* ON *)
  SYSTEM_CALL [CAMERA] 'SONCA00P' (CAMERA,1,0,0)
[TP,26] = [CAMERA,249]

PUSH[TP,27]                       (* OFF *)
  SYSTEM_CALL [CAMERA] 'SONCA00P' (CAMERA,0,0,0)
[TP,27] = (![CAMERA,249])

(*************************************)
(* PRESET METHOD #2 - 1 TO 128 PRESETS *)
(*************************************
SYSTEM_CALL [CAMERA] 'SONCA001' (CAMERA,TP,4,3,1,2,5,6,7,8,9,10)
SYSTEM_CALL [CAMERA] 'SONCA003' (CAMERA,TP,22,23,24,25,0,0)
SYSTEM_CALL [CAMERA] 'SONCA004' (CAMERA,TP,29,30,32,31)
```

```
SYSTEM_CALL [CAMERA] 'SONCA00B' (CAMERA,CAMERA_BUFF)

PUSH[TP,1]                          (* TILT UP *)
PUSH[TP,2]                          (* TILT DN *)
PUSH[TP,3]                          (* PAN RIGHT *)
PUSH[TP,4]                          (* PAN LEFT *)
PUSH[TP,5]                          (* ZOOM TELE *)
PUSH[TP,6]                          (* ZOOM WIDE *)
PUSH[TP,7]                          (* FOCUS NEAR *)
PUSH[TP,8]                          (* FOCUS_FAR *)
  CAM_PRESET = 0 (* CLEAR PRESET FEEDBACK IF WE MOVE CAMERA! *)

PUSH[TP,40]                         (* SAVE PRESET *)
  PRESET_SAVE = !PRESET_SAVE
[TP,40] = (PRESET_SAVE)

PUSH[TP,41]                         (* PRESET 1 *)
PUSH[TP,42]                         (* PRESET 2 *)
PUSH[TP,43]                         (* PRESET 3 *)
PUSH[TP,44]                         (* PRESET 4 *)
PUSH[TP,45]                         (* PRESET 5 *)
PUSH[TP,46]                         (* PRESET 6 *)
PUSH[TP,47]                         (* PRESET 7 *)
PUSH[TP,48]                         (* PRESET 8 *)
PUSH[TP,49]                         (* PRESET 9 *)
PUSH[TP,50]                         (* PRESET 10 *)
PUSH[TP,51]                         (* PRESET 11 *)
PUSH[TP,52]                         (* PRESET 12 *)
PUSH[TP,53]                         (* PRESET 13 *)
PUSH[TP,54]                         (* PRESET 14 *)
PUSH[TP,55]                         (* PRESET 15 *)
PUSH[TP,56]                         (* PRESET 16 *)
PUSH[TP,57]                         (* PRESET 17 *)
PUSH[TP,58]                         (* PRESET 18 *)
PUSH[TP,59]                         (* PRESET 19 *)
PUSH[TP,60]                         (* PRESET 20 *)
{
  CAM_PRESET = PUSH_CHANNEL - 40 (* 1-20 *)
  IF (PRESET_SAVE)
  {
    PRESET_FUNC = CAM_PRESET + 100 + 256
    SYSTEM_CALL [CAMERA] 'SONCA00F' (CAMERA,PRESET_FUNC)
  }
  ELSE
  {
    PRESET_FUNC = CAM_PRESET + 100
    SYSTEM_CALL [CAMERA] 'SONCA00F' (CAMERA,PRESET_FUNC)
  }
  OFF[PRESET_SAVE]
}
[TP,41] = (CAM_PRESET = 1)
[TP,42] = (CAM_PRESET = 2)
[TP,43] = (CAM_PRESET = 3)
[TP,44] = (CAM_PRESET = 4)
[TP,45] = (CAM_PRESET = 5)
[TP,46] = (CAM_PRESET = 6)
[TP,47] = (CAM_PRESET = 7)
```

```
[TP,48] = (CAM_PRESET = 8)
[TP,49] = (CAM_PRESET = 9)
[TP,50] = (CAM_PRESET = 10)
[TP,51] = (CAM_PRESET = 11)
[TP,52] = (CAM_PRESET = 12)
[TP,53] = (CAM_PRESET = 13)
[TP,54] = (CAM_PRESET = 14)
[TP,55] = (CAM_PRESET = 15)
[TP,56] = (CAM_PRESET = 16)
[TP,57] = (CAM_PRESET = 17)
[TP,58] = (CAM_PRESET = 18)
[TP,59] = (CAM_PRESET = 19)
[TP,60] = (CAM_PRESET = 20)
*)

(*********************************************************)
(*                     END OF PROGRAM                    *)
(*        DO NOT PUT ANY CODE BELOW THIS COMMENT         *)
(*********************************************************)
```

XXVI. Parkervision Camera System II - Version 2.01 ----------

```
*****************************************************************
*******   Instancing is required on this class of calls! **********
*****************************************************************
```

A. I. SYSTEM_CALL [CAM] 'PARCA001' (CAM,PANEL,PLR,PRB,TUB,TDB,ZTB,ZWB,FNB,FFB,AFB,MFB)

Basic camera control. Parameters are:
Pan Left, Pan Right, Tilt Up, Tilt Down, Zoom Tele, Zoom Wide,
Focus Near, Focus Far, Auto Focus, Manual Focus.

B. II. SYSTEM_CALL [CAM] 'PARCA002' (CAM,PANEL,P1B,P2B,P3B,P4B,P5B,P6B,P7B,P8B,P9B,P10B,SPB)

Basic camera presets. Parameters are:
Presets 1-10 and Save. If 0 is passed for save button, preset
buttons will store presets when held for 2 seconds. Preset buttons
will recall only if 256 is passed for SPB.

C. III. SYSTEM_CALL [CAM] 'PARCA003' (CAM,PANEL,IOB,ICB,AIB,MIB,AGBONB,AGCOFFB)

Basic iris control. Parameters are:
Iris Open, Iris Close, Auto Iris, Manual Iris, AGC on, AGC off.
AGC on and AGC off are not implemented.

D. IV. SYSTEM_CALL [CAM] 'PARCA007' (CAM,PANEL,SLR,SRB,SUB,SDB)

Autotrack subject adjust. Parameters are:
Subject Left, Subject Right, Subject Up, Subject Down.

E. V. SYSTEM_CALL [CAM] 'PARCA008' (CAM,PANEL,ATRACKON,ATRACKOFF,P1B,P2B,P3B,P4B,SPB)

Autotrack presets. Parameters are:
Autotrack on, Autotrack off, Tight Presets, Wide Preset, Right Preset.
Left Preset, and Save. If 0 is passed for save button, preset
buttons will store presets for 2 seconds. Preset buttons
will only save if 256 is passed for SPB.

F. VI. SYSTEM_CALL [CAM] 'PARCA00F' (CAM,PARAMETER)

Camera function call. Any of the following cam be passed
as PARAMETER to achieve a desired camera function.

PAN_RIGHT	= 31
TILT_DN	= 32
PAN_LEFT	= 35
TILT_UP	= 36
ZOOM_TELE	= 3
FOCUS_NEAR	= 4
ZOOM_WIDE	= 7
FOCUS_FAR	= 8

```
        PAN_STOP                        = 287          (* PAN RIGHT | $100 *)
        TILT_STOP                       = 288          (* TILT DN | $100 *)
        ZOOM_STOP                       = 259          (* ZOOM TELE | $100 *)
        FOCUS_STOP                      = 260          (* FOCUS NEAR | $100 *)
        AUTO_FOCUS                      = 10
        MAN_FOCUS                       = 11
        HOME_POSITION                   = 99
        PRESET1                         = 101
        PRESET2                         = 102
        PRESET3                         = 103
        PRESET4                         = 104
        PRESET5                         = 105
        PRESET6                         = 106
        PRESET7                         = 107
        PRESET8                         = 108
        PRESET9                         = 109
        PRESET10                        = 110
        SAVE PRESET1                    = 357
        SAVE PRESET2                    = 358
        SAVE PRESET3                    = 359
        SAVE PRESET4                    = 360
        SAVE PRESET5                    = 361
        SAVE PRESET6                    = 362
        SAVE PRESET7                    = 363
        SAVE PRESET8                    = 364
        SAVE PRESET9                    = 365
        SAVE PRESET10                   = 366
        IRIS_OPEN                       = 1
        IRIS_CLOSE                      = 2
        AUTO_IRIS                       = 61
        MAN_IRIS                        = 62
        IRIS_STOP                       = 257          (* IRIS OPEN | $100 *)
        PARCA000_ATRACK                 = 79
        PARCA000_ATRACK_OFF             = 80
        PARCA000_ATRACK_TIGHT           = 81
        PARCA000_ATRACK_WIDE            = 82
        PARCA000_ATRACK_RIGHT           = 83
        PARCA000_ATRACK_LEFT            = 84
        PARCA000_SAVE_ATRACK_TIGHT      = 337
        PARCA000_SAVE_ATRACK_WIDE       = 338
        PARCA000_SAVE_ATRACK_RIGHT      = 339
        PARCA000_SAVE_ATRACK_LEFT       = 340
        PARCA000_SUB_UP                 = 86
        PARCA000_SUB_DN                 = 87
        PARCA000_SUB_RIGHT              = 88
        PARCA000_SUB_LEFT               = 89
        PARCA000_SUB_STOP               = 342       (* SUB UP | $100 *)

Additional presets are available by passing the following
parameters to the function call:
        PARCA000_PRESET_BEG             = 111       (* PRESETS 11 *)
           ... THRU ...
        PARCA000_PRESET_END             = 228       (* MAX 128 PRESETS *)

        PARCA000_SAVE_PRESET_BEG        = 367       (* PRESETS 11 *)
           ... THRU ...
        PARCA000_SAVE_PRESET_END        = 484       (* MAX 128 PRESETS *)
```

```
The following constants can be overridden:
PARCA000_PSET_HOLD_TIME = 20 (* 2 SECONDS *)
  Defines how long the preset button is held before a preset
  is stored when a 0 had been passed for the SPB for call PARCA002.

PARCA001_DEFEAT_FEEDBACK = 0
PARCA002_DEFEAT_FEEDBACK = 0
PARCA003_DEFEAT_FEEDBACK = 0
PARCA007_DEFEAT_FEEDBACK = 0
PARCA008_DEFEAT_FEEDBACK = 0
  Defeats feedback for each of the calls listed when the constant
  is defined as non-zero (usually 1)

PARCA000_AMX_ADDR        = $FB (* STANDARD AMX ADDRESS FOR CAMERAMAN *)
PARCA000_CAM_ADDR        = $04 (* DIPSWITCH SET TO 1 *)
  Re-defines the default address.  The AMX address should not be
  modified.  The Parkervision address ca be modified to any
valid dipswitch setting. The address is calculated by the
  formula: ADDRESS = (DIPSWITCH+1) * 4.

PARCA000_PAN_SPEED        = $0D            (* 1-$18, $18 IS MAX *)
PARCA000_TILT_SPEED       = $0A            (* 1-$14, $14 IS MAX *)
PARCA000_ZOOM_SPEED       = $05            (* 1-8, 8 IS MAX *)
  Re-defines the default speeds for the camera operations
  when the button passed to PARCA001 is pushed.
```

G. EXAMPLE PROGRAM:

```
(***********************************************************)
(*                DEVICE DEFINITIONS GO BELOW              *)
(***********************************************************)
DEFINE_DEVICE

CAMERA              = 1
TP                  = 128

(***********************************************************)
(*               CONSTANT DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_CONSTANT

PARCA000_MAX_PRESETS      = 30

(***********************************************************)
```

```
(*               VARIABLE DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_VARIABLE

CAM_PRESET                          (* CURRENT CAMERA PRESET *)
PRESET_SAVE                         (* 1 TO SAVE *)
PRESET_FUNC                         (* PRESET FUNCTION NUMBER *)

(***********************************************************)
(*             SUBROUTINE DEFINITIONS GO BELOW             *)
(***********************************************************)

(***********************************************************)
(*              BEGIN START SECTION BELOW                  *)
(***********************************************************)
DEFINE_START

(***********************************************************)
(*             BEGIN PROGRAM SECTION BELOW                 *)
(***********************************************************)
DEFINE_PROGRAM

(*************************************)
(* PRESET METHOD #1 - 1 TO 10 PRESETS  *)
(*************************************)
SYSTEM_CALL [CAMERA] 'PARCA001' (CAMERA,TP,4,3,1,2,5,6,7,8,9,10)
SYSTEM_CALL [CAMERA] 'PARCA002' (CAMERA,TP,11,12,13,14,15,16,17,18,19,20,0)
SYSTEM_CALL [CAMERA] 'PARCA003' (CAMERA,TP,22,23,24,25,0,0)
SYSTEM_CALL [CAMERA] 'PARCA007' (CAMERA,TP,32,31,29,30)
SYSTEM_CALL [CAMERA] 'PARCA008' (CAMERA,TP,38,39,33,34,35,36,37)

(*************************************)
(* PRESET METHOD #2 - 1 TO 128 PRESETS *)
(*************************************
SYSTEM_CALL [CAMERA] 'PARCA001' (CAMERA,TP,4,3,1,2,5,6,7,8,9,10)
SYSTEM_CALL [CAMERA] 'PARCA003' (CAMERA,TP,22,23,24,25,0,0)
SYSTEM_CALL [CAMERA] 'PARCA007' (CAMERA,TP,32,31,29,30)
SYSTEM_CALL [CAMERA] 'PARCA008' (CAMERA,TP,38,39,33,34,35,36,37)

PUSH[TP,1]                          (* TILT UP *)
PUSH[TP,2]                          (* TILT DN *)
PUSH[TP,3]                          (* PAN RIGHT *)
PUSH[TP,4]                          (* PAN LEFT *)
PUSH[TP,5]                          (* ZOOM TELE *)
PUSH[TP,6]                          (* ZOOM WIDE *)
PUSH[TP,7]                          (* FOCUS NEAR *)
PUSH[TP,8]                          (* FOCUS_FAR *)
  CAM_PRESET = 0 (* CLEAR PRESET FEEDBACK IF WE MOVE CAMERA! *)

PUSH[TP,40]                         (* SAVE PRESET *)
  PRESET_SAVE = !PRESET_SAVE
[TP,40] = (PRESET_SAVE)

PUSH[TP,41]                         (* PRESET 1 *)
PUSH[TP,42]                         (* PRESET 2 *)
PUSH[TP,43]                         (* PRESET 3 *)
PUSH[TP,44]                         (* PRESET 4 *)
```

```
PUSH[TP,45]                          (* PRESET 5 *)
PUSH[TP,46]                          (* PRESET 6 *)
PUSH[TP,47]                          (* PRESET 7 *)
PUSH[TP,48]                          (* PRESET 8 *)
PUSH[TP,49]                          (* PRESET 9 *)
PUSH[TP,50]                          (* PRESET 10 *)
PUSH[TP,51]                          (* PRESET 11 *)
PUSH[TP,52]                          (* PRESET 12 *)
PUSH[TP,53]                          (* PRESET 13 *)
PUSH[TP,54]                          (* PRESET 14 *)
PUSH[TP,55]                          (* PRESET 15 *)
PUSH[TP,56]                          (* PRESET 16 *)
PUSH[TP,57]                          (* PRESET 17 *)
PUSH[TP,58]                          (* PRESET 18 *)
PUSH[TP,59]                          (* PRESET 19 *)
PUSH[TP,60]                          (* PRESET 20 *)
{
  CAM_PRESET = PUSH_CHANNEL - 40 (* 1-20 *)
  IF (PRESET_SAVE)
  {
    PRESET_FUNC = CAM_PRESET + 100 + 256
    SYSTEM_CALL [CAMERA] 'PARCA00F' (CAMERA,PRESET_FUNC)
  }
  ELSE
  {
    PRESET_FUNC = CAM_PRESET + 100
    SYSTEM_CALL [CAMERA] 'PARCA00F' (CAMERA,PRESET_FUNC)
  }
  OFF[PRESET_SAVE]
}
[TP,41] = (CAM_PRESET = 1)
[TP,42] = (CAM_PRESET = 2)
[TP,43] = (CAM_PRESET = 3)
[TP,44] = (CAM_PRESET = 4)
[TP,45] = (CAM_PRESET = 5)
[TP,46] = (CAM_PRESET = 6)
[TP,47] = (CAM_PRESET = 7)
[TP,48] = (CAM_PRESET = 8)
[TP,49] = (CAM_PRESET = 9)
[TP,50] = (CAM_PRESET = 10)
[TP,51] = (CAM_PRESET = 11)
[TP,52] = (CAM_PRESET = 12)
[TP,53] = (CAM_PRESET = 13)
[TP,54] = (CAM_PRESET = 14)
[TP,55] = (CAM_PRESET = 15)
[TP,56] = (CAM_PRESET = 16)
[TP,57] = (CAM_PRESET = 17)
[TP,58] = (CAM_PRESET = 18)
[TP,59] = (CAM_PRESET = 19)
[TP,60] = (CAM_PRESET = 20)
*)
(***********************************************************)
(*                     END OF PROGRAM                      *)
(*         DO NOT PUT ANY CODE BELOW THIS COMMENT          *)
(***********************************************************)
```

XXVII. Canon VCC3 calls - Version 2.04 ----------

```
*****************************************************************
*******   Instancing is required on this class of calls! **********
*****************************************************************
```

A. I. SYSTEM_CALL [CAM] 'CANCA000' (CAM)

```
Initialization call.  This call should always be included
at start up.  It initializes the camera and the system call.
```

B. II. SYSTEM_CALL [CAM] 'CANCA001' (CAM,PANEL,PLR,PRB,TUB,TDB,ZTB,ZWB,FNB,FFB,AFB,MFB)

```
Basic camera control.  Parameters are:
Pan Left, Pan Right, Tilt Up, Tilt Down, Zoom Tele, Zoom Wide,
Focus Near, Focus Far, Auto Focus, Manual Focus.
```

C. III. SYSTEM_CALL [CAM] 'CANCA002' (CAM,PANEL,P1B,P2B,P3B,P4B,P5B,P6B,P7B,P8B,P9B,P10B,SPB)

```
Basic camera presets.  Parameters are:
Presets 1-10 and Save.  If 0 is passed for save button, preset
buttons will store presets when held for 2 seconds.  Preset buttons
will recall only if 256 is passed for SPB.
```

D. IV. SYSTEM_CALL [CAM] 'CANCA003' (CAM,PANEL,IOB,ICB,AIB,MIB,AGBONB,AGCOFFB)

```
Basic iris control. Parameters are:
Iris Open, Iris Close, Auto Iris, Manual Iris, AGC on, AGC off.
```

E. V. SYSTEM_CALL [CAM] 'CANCA004' (CAM,PANEL,SOB,SCB,ASB,MSB)

```
Basic shutter control. Parameters are:
Shutter Open, Shutter Close, Auto Shutter, Manual Shutter.
```

F. VI. SYSTEM_CALL [CAM] 'CANCA00B' (CAM,CAM_BUFFER)

```
Buffer processing.  This call should always be included
in mainline.  It communicates with the camera.
The CAMERA_BUFFER is a variable that must be created
in your DEFINE_VARIABLE section and you must assign it to
be a buffer in DEFINE_START with a line like:
CREATE_BUFFER CAM,CAM_BUFFER
```

G. VII. SYSTEM_CALL [CAM] 'CANCA00F' (CAM,PARAMETER)

```
Camera function call.  Any of the following cam be passed
as PARAMETER to achieve a desired camera function.
       PAN_RIGHT                       = 31
       TILT_DN                         = 32
       PAN_LEFT                        = 35
```

```
        TILT_UP                         = 36
        ZOOM_TELE                       = 3
        FOCUS_NEAR                      = 4
        ZOOM_WIDE                       = 7
        FOCUS_FAR                       = 8
        PAN_STOP                        = 287          (* PAN RIGHT | $100 *)
        TILT_STOP                       = 288          (* TILT DN | $100 *)
        ZOOM_STOP                       = 259          (* ZOOM TELE | $100 *)
        FOCUS_STOP                      = 260          (* FOCUS NEAR | $100 *)
        AUTO_FOCUS                      = 10
        MAN_FOCUS                       = 11
        HOME_POSITION                   = 99
        PRESET1                         = 101
        PRESET2                         = 102
        PRESET3                         = 103
        PRESET4                         = 104
        PRESET5                         = 105
        PRESET6                         = 106
        PRESET7                         = 107
        PRESET8                         = 108
        PRESET9                         = 109
        PRESET10                        = 110
        SAVE PRESET1                    = 357
        SAVE PRESET2                    = 358
        SAVE PRESET3                    = 359
        SAVE PRESET4                    = 360
        SAVE PRESET5                    = 361
        SAVE PRESET6                    = 362
        SAVE PRESET7                    = 363
        SAVE PRESET8                    = 364
        SAVE PRESET9                    = 365
        SAVE PRESET10                   = 366
        IRIS_OPEN                       = 1
        IRIS_CLOSE                      = 2
        AUTO_IRIS                       = 61
        MAN_IRIS                        = 62
        IRIS_STOP                       = 257          (* IRIS OPEN | $100 *)
        SHUTTER_OPEN                    = 65
        SHUTTER_CLOSE                   = 66
        AUTO_SHUTTER                    = 67
        MAN_SHUTTER                     = 68
        SHUTTER_STOP                    = 270          (* SHUTTER OPEN | $100 *)
        CANCA000_FADE_OUT               = 71
        CANCA000_FADE_IN                = 72
        CANCA000_VIDEO_MUTE_ON          = 73
        CANCA000_VIDEO_MUTE_OFF         = 74
        CANCA000_AUTO_EXPOSE_MODE       = 80
        CANCA000_MAN_EXPOSE_MODE        = 81
        CANCA000_FULL_AUTO_AE           = 82
        CANCA000_PC_CONTROL_MODE        = 83
        CANCA000_IR_CONTROL_MODE        = 84
        CANCA000_RESET_CAMERA           = 88
        CANCA000_INIT_CAMERA            = 89

Additional presets are available by passing the following
parameters to the function call:
        CANCA000_PRESET_BEG             = 111       (* PRESETS 11 *)
```

```
      ... THRU ...
    CANCA000_PRESET_END             = 130       (* MAX 30 PRESETS *)

    CANCA000_SAVE_PRESET_BEG        = 367       (* PRESETS 11 *)
      ... THRU ...
    CANCA000_SAVE_PRESET_END        = 386       (* MAX 30 PRESETS *)
```

However, the maximum number of presets is determined by the constant:

```
    CANCA000_MAX_PRESETS             = 30
```

which can be overridden to any value between 1 and 255. Do not us
0 and be aware that setting it to a value of less than 10 will cause
buttons 7, 8, 9, or 10 in system call CANCA002 not to work.

The following constants can be overridden:

```
CANCA000_PSET_HOLD_TIME = 20 (* 2 SECONDS *)
```

Defines how long the preset button is held before a preset is stored when a 0 had been passed for the SPB for call CANCA002.

```
CANCA001_DEFEAT_FEEDBACK = 0
CANCA002_DEFEAT_FEEDBACK = 0
CANCA003_DEFEAT_FEEDBACK = 0
CANCA004_DEFEAT_FEEDBACK = 0
```

Defeats feedback for each of the calls listed when the constant is defined as non-zero (usually 1)

```
CANCA000_PAN_SPEED        = $18             (* 1-$4C, $4C IS MAX *)
CANCA000_TILT_SPEED       = $12             (* 1-$46, $46 IS MAX *)
CANCA000_ZOOM_SPEED       = $03             (* 1-7, 7 IS MAX *)
CANCA000_FOCUS_SPEED      = $03             (* 1-7, 7 IS MAX *)
```

Re-defines the default speeds for the camera operations when the button passed to CANCA001 is pushed.

H. EXAMPLE PROGRAM:

```
(***********************************************************)
(*                 DEVICE DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_DEVICE

CAMERA              = 1
TP                  = 128

(***********************************************************)
(*               CONSTANT DEFINITIONS GO BELOW             *)
(***********************************************************)
DEFINE_CONSTANT
```

```
CANCA000_MAX_PRESETS      = 30

(***********************************************************)
(*             VARIABLE DEFINITIONS GO BELOW               *)
(***********************************************************)
DEFINE_VARIABLE

CAMERA_BUFF[255]                  (* INCOMING BUFFER *)
CAM_PRESET                        (* CURRENT CAMERA PRESET *)
PRESET_SAVE                       (* 1 TO SAVE *)
PRESET_FUNC                       (* PRESET FUNCTION NUMBER *)

(***********************************************************)
(*             SUBROUTINE DEFINITIONS GO BELOW             *)
(***********************************************************)

(***********************************************************)
(*                   START SECTION GO BELOW                *)
(***********************************************************)
DEFINE_START

CREATE_BUFFER CAMERA,CAMERA_BUFF
SYSTEM_CALL [CAMERA] 'CANCA000' (CAMERA)

(***********************************************************)
(*                 PROGRAM SECTION GO BELOW                *)
(***********************************************************)
DEFINE_PROGRAM

(*************************************)
(* PRESET METHOD #1 - 1 TO 10 PRESETS  *)
(*************************************)
SYSTEM_CALL [CAMERA] 'CANCA001' (CAMERA,TP,4,3,1,2,5,6,7,8,9,10)
SYSTEM_CALL [CAMERA] 'CANCA002' (CAMERA,TP,11,12,13,14,15,16,17,18,19,20,21)
SYSTEM_CALL [CAMERA] 'CANCA003' (CAMERA,TP,22,23,24,25,0,0)
SYSTEM_CALL [CAMERA] 'CANCA004' (CAMERA,TP,29,30,32,31)
SYSTEM_CALL [CAMERA] 'CANCA00B' (CAMERA,CAMERA_BUFF)

(*************************************)
(* PRESET METHOD #2 - 1 TO 128 PRESETS *)
(*************************************
SYSTEM_CALL [CAMERA] 'CANCA001' (CAMERA,TP,4,3,1,2,5,6,7,8,9,10)
SYSTEM_CALL [CAMERA] 'CANCA003' (CAMERA,TP,22,23,24,25,0,0)
SYSTEM_CALL [CAMERA] 'CANCA004' (CAMERA,TP,29,30,32,31)
SYSTEM_CALL [CAMERA] 'CANCA00B' (CAMERA,CAMERA_BUFF)

PUSH[TP,1]                        (* TILT UP *)
PUSH[TP,2]                        (* TILT DN *)
PUSH[TP,3]                        (* PAN RIGHT *)
PUSH[TP,4]                        (* PAN LEFT *)
PUSH[TP,5]                        (* ZOOM TELE *)
PUSH[TP,6]                        (* ZOOM WIDE *)
PUSH[TP,7]                        (* FOCUS NEAR *)
PUSH[TP,8]                        (* FOCUS_FAR *)
  CAM_PRESET = 0 (* CLEAR PRESET FEEDBACK IF WE MOVE CAMERA! *)

PUSH[TP,40]                       (* SAVE PRESET *)
```

```
  PRESET_SAVE = !PRESET_SAVE
[TP,40] = (PRESET_SAVE)

PUSH[TP,41]                         (* PRESET 1 *)
PUSH[TP,42]                         (* PRESET 2 *)
PUSH[TP,43]                         (* PRESET 3 *)
PUSH[TP,44]                         (* PRESET 4 *)
PUSH[TP,45]                         (* PRESET 5 *)
PUSH[TP,46]                         (* PRESET 6 *)
PUSH[TP,47]                         (* PRESET 7 *)
PUSH[TP,48]                         (* PRESET 8 *)
PUSH[TP,49]                         (* PRESET 9 *)
PUSH[TP,50]                         (* PRESET 10 *)
PUSH[TP,51]                         (* PRESET 11 *)
PUSH[TP,52]                         (* PRESET 12 *)
PUSH[TP,53]                         (* PRESET 13 *)
PUSH[TP,54]                         (* PRESET 14 *)
PUSH[TP,55]                         (* PRESET 15 *)
PUSH[TP,56]                         (* PRESET 16 *)
PUSH[TP,57]                         (* PRESET 17 *)
PUSH[TP,58]                         (* PRESET 18 *)
PUSH[TP,59]                         (* PRESET 19 *)
PUSH[TP,60]                         (* PRESET 20 *)
{
  CAM_PRESET = PUSH_CHANNEL - 40 (* 1-20 *)
  IF (PRESET_SAVE)
  {
    PRESET_FUNC = CAM_PRESET + 100 + 256
    SYSTEM_CALL [CAMERA] 'CANCA00F' (CAMERA,PRESET_FUNC)
  }
  ELSE
  {
    PRESET_FUNC = CAM_PRESET + 100
    SYSTEM_CALL [CAMERA] 'CANCA00F' (CAMERA,PRESET_FUNC)
  }
  OFF[PRESET_SAVE]
}
[TP,41] = (CAM_PRESET = 1)
[TP,42] = (CAM_PRESET = 2)
[TP,43] = (CAM_PRESET = 3)
[TP,44] = (CAM_PRESET = 4)
[TP,45] = (CAM_PRESET = 5)
[TP,46] = (CAM_PRESET = 6)
[TP,47] = (CAM_PRESET = 7)
[TP,48] = (CAM_PRESET = 8)
[TP,49] = (CAM_PRESET = 9)
[TP,50] = (CAM_PRESET = 10)
[TP,51] = (CAM_PRESET = 11)
[TP,52] = (CAM_PRESET = 12)
[TP,53] = (CAM_PRESET = 13)
[TP,54] = (CAM_PRESET = 14)
[TP,55] = (CAM_PRESET = 15)
[TP,56] = (CAM_PRESET = 16)
[TP,57] = (CAM_PRESET = 17)
[TP,58] = (CAM_PRESET = 18)
[TP,59] = (CAM_PRESET = 19)
[TP,60] = (CAM_PRESET = 20)
```

```
*)
(*************************************************************)
(*                      END OF PROGRAM                       *)
(*         DO NOT PUT ANY CODE BELOW THIS COMMENT            *)
(*************************************************************)
```

End of document