PIDP10

Paul Koning logo https://nedbatchelder.com/files/declogo11.svg

19th October 2025

Installation

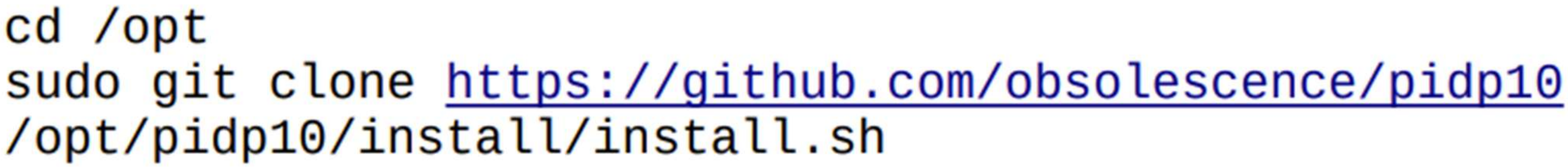
DOCUMENTATION

- PiDP10 user manual: https://obsolescence.dev/pidp10-sw/PiDP-10_Manual.pdf
- ITS user manual: https://obsolescenceguaranteed.blogspot.com/2020/02/a-turists-guide-to-its.html
- Primer on ITS: https://github.com/PDP-10/its/blob/master/doc/_info_/its.primer
- Wiki: https://github.com/larsbrinkhoff/its-manual/wiki
- Emacs: https://www.gnu.org/software/emacs/refcards/pdf/refcard.pdf
- PDP-10 Assembler: http://pdp10.nocrew.org/docs/instruction-set/pdp-10.html

INSTALLING PIDP10

https://obsolescence.dev/pidp10-sw/PiDP-10_Manual.pdf

- The simulation, **simh**, requires the 64-bit version of the Pi OS operating system
- Download the PiDP10 software
- The installer asks a lot of questions. to which the answer is ‘y’
- But there is no need to download the source code



```
cd /opt
sudo git clone https://github.com/obsolescence/pidp10
/opt/pidp10/install/install.sh
```

INSTALLING ON X86 LINUX

https://obsolescence.dev/pidp10-sw/PiDP-10_Manual.pdf

- Install the same software on the x86 Linux system
- Say N to the following questions:
  - Automatically start the PiDP-10 core when logging in?
  - Add a DEC flavour to the Pi's desktop?
  - Reduce telnet release time?
- Say Y to the following question:
  - Install required dependencies for running the PiDP-10?
- After installation, rebuild for x86
  - **cd /opt/pidp10/bin**
  - **tar xzvf binaries-for-x86-64.tar.gz**

Using ITS

STARTING ITS ON PIDP10

- When the Raspberry Pi starts, **pdpcontrol start** runs
- The front panel is read to decide which operating system to boot
- With no front panel, in the Raspberry Pi terminal, run **pdpcontrol stop**, wait for the simulation to stop, then run **pdpcontrol start *X***, where *X* is the number of the system to boot
- For ITS, use **pdpcontrol start 1**

Command	Meaning
pdpcontrol stop	Stop PiDP10
pdpcontrol start	Start PiDP10, using the front panel to select the operating system
pdpcontrol start N	Start PiDP10: 0=BLINKY 1= ITS 2= TOPS-10 200= TOPS-20

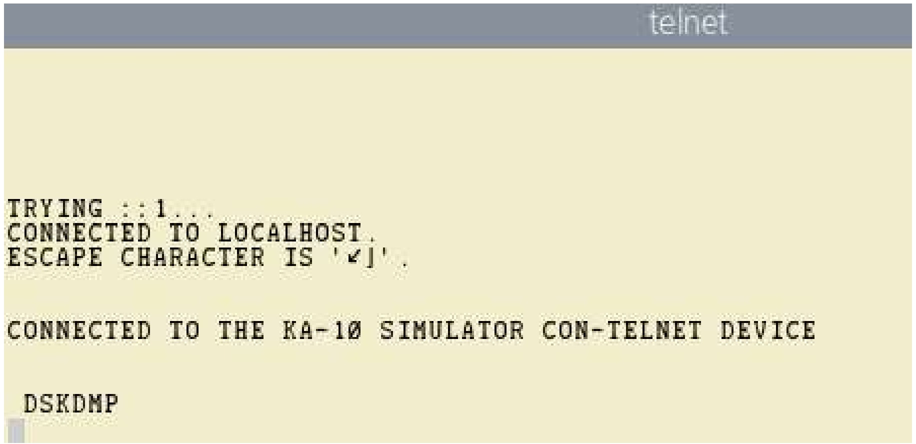
STARTING TERMINAL

- Terminals are started using **pdp *terminal***, or by double clicking on the icon on the left-hand side of the screen

Command	Meaning
pdp con	Starts teleprinter
pdp telcon	Starts telnet
pdp vt52	Starts VT52 terminal
pdp tvcon	Starts Knight TV graphical terminal

TELEPRINTER AND TELNET

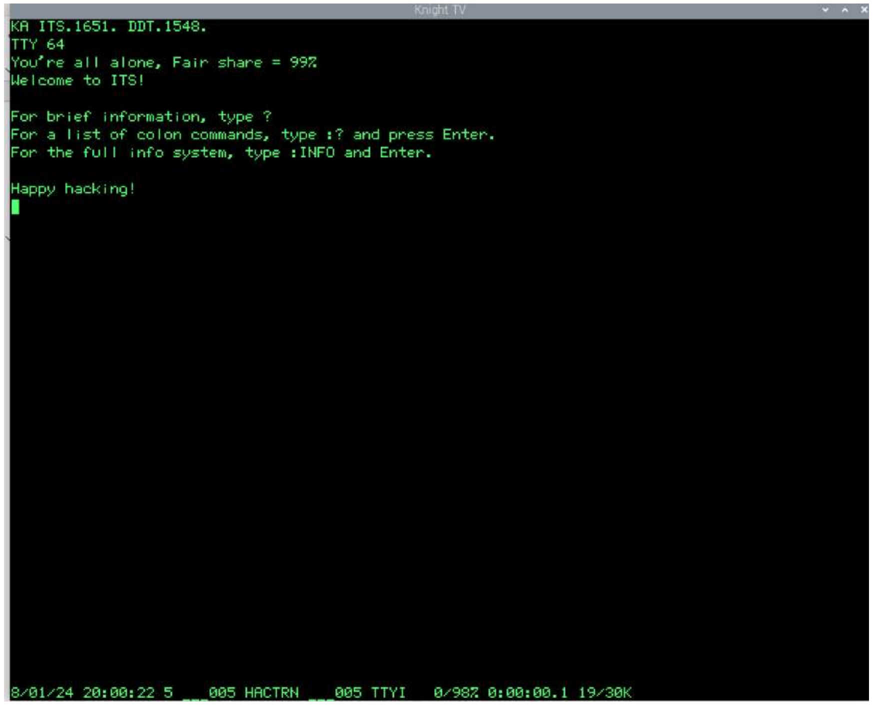
- Using **pdp con** or **pdp telcon**, or the CON or TELCON icons on the left-hand side, start a terminal
- (Restarting using **pdpcontrol stop** closes the terminal)
- Wait until the DSKDMP prompt appears
- Boot ITS using **its<return><esc>G**
- Booting message appear in the teleprinter or telnet windows



```
TRYING ::1...
CONNECTED TO LOCALHOST.
ESCAPE CHARACTER IS '↙]'.


CONNECTED TO THE KA-10 SIMULATOR CON-TELNET DEVICE


 DSKDMP
```

KNIGHT TV

- Using **pdp tvcon** or the KNIGHT icon on the left side of the screen
- Once ITS is running, login using the **CALL** sequence, **F1**, the terminal had a special CALL key



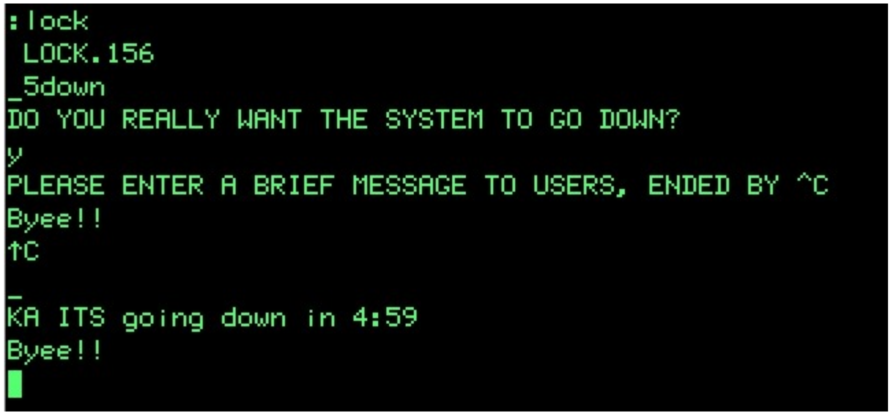
```
KA ITS.1651. DDT.1548.
TTY 64
You're all alone, Fair share = 99%
Welcome to ITS!

For brief information, type ?
For a list of colon commands, type :? and press Enter.
For the full info system, type :INFO and Enter.

Happy hacking!
█
```

```
8/01/24 20:00:22 5    005 HACTRN    005 TTYI   0/98% 0:00:00.1 19/30K
```

STOPPING PDP-10

https://obsolescenceguaranteed.blogspot.com/2020/02/a-turists-guide-to-its.html

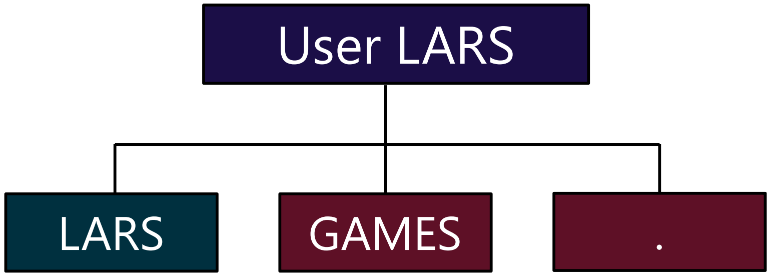
- Type **:LOCK**
- Type **5DOWN** against the _ prompt
  - The number is the number of minutes to wait
- ITS will ask if you want to halt the system, answer **Y**
- Type a message and press **Ctrl-C**
- If you are logged out then the system goes down immediately
- If you're still logged on, press **Q** and then **:LOGOUT**



```
:lock
 LOCK.156
_5down
DO YOU REALLY WANT THE SYSTEM TO GO DOWN?
y
PLEASE ENTER A BRIEF MESSAGE TO USERS, ENDED BY ^C
Byee!!
↑C

_
KA ITS going down in 4:59
Byee!!
█
```

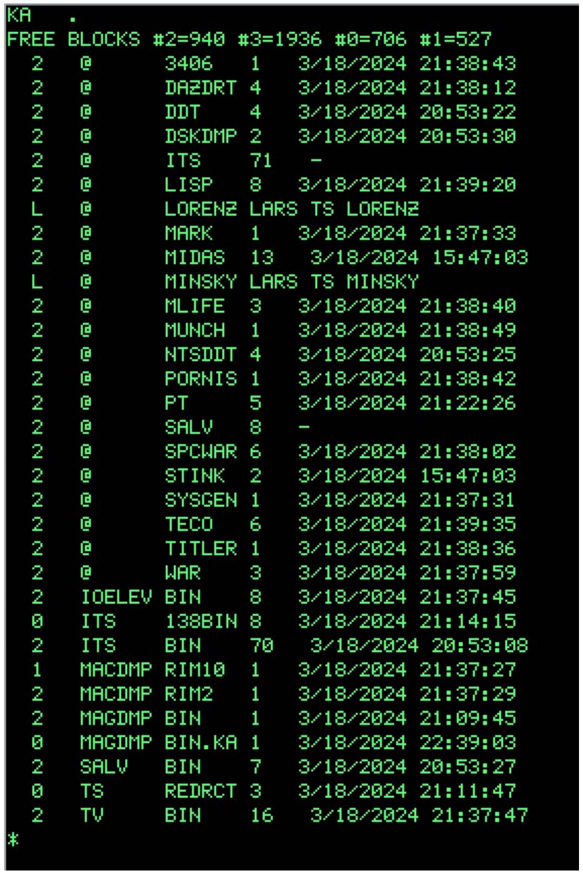
FOLDERS

- A user account has folders under a root folder
- There is only one layer of folders
- The user has a folder with their name, which is the default folder when they log on
- The current working folder is shown at the bottom of the Knight terminal
- All folders are available to all users



SYSTEM FOLDER

- The **.** folder is the system folder
- It contains the operating system **@ ITS**, as well as standalone programs like Lisp, **@ LISP**, and timeshare programs like **TS LISP**



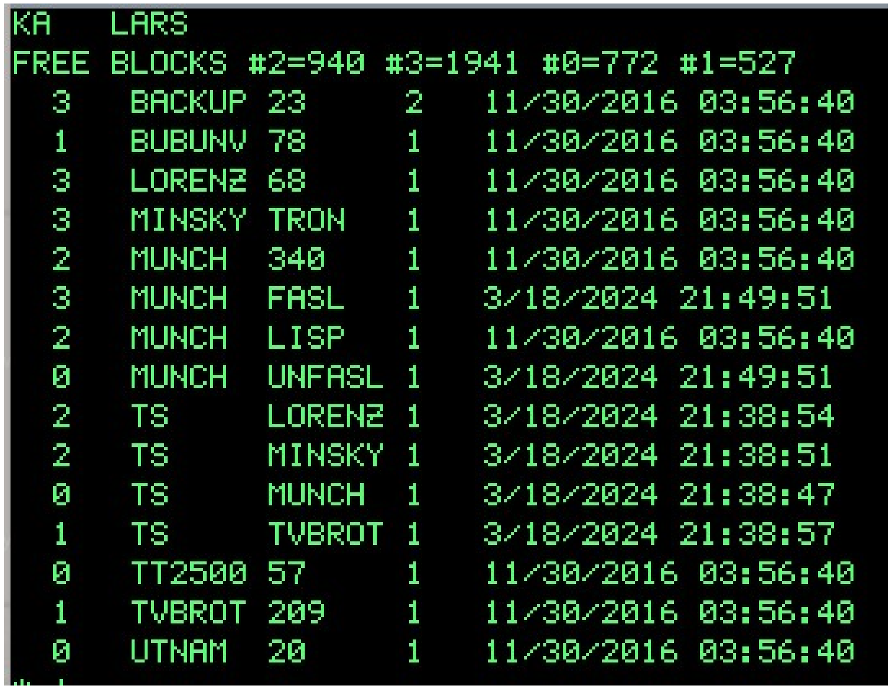
```
KA   .
FREE BLOCKS #2=940 #3=1936 #0=706 #1=527
  2   @      3406   1   3/18/2024 21:38:43
  2   @      DAZDRT 4   3/18/2024 21:38:12
  2   @      DDT    4   3/18/2024 20:53:22
  2   @      DSKDMP 2   3/18/2024 20:53:30
  2   @      ITS    71   -
  2   @      LISP   8   3/18/2024 21:39:20
  L   @      LORENZ LARS TS LORENZ
  2   @      MARK   1   3/18/2024 21:37:33
  2   @      MIDAS  13   3/18/2024 15:47:03
  L   @      MINSKY LARS TS MINSKY
  2   @      MLIFE  3   3/18/2024 21:38:40
  2   @      MUNCH  1   3/18/2024 21:38:49
  2   @      NTSDDT 4   3/18/2024 20:53:25
  2   @      PORNIS 1   3/18/2024 21:38:42
  2   @      PT     5   3/18/2024 21:22:26
  2   @      SALV   8   -
  2   @      SPCWAR 6   3/18/2024 21:38:02
  2   @      STINK  2   3/18/2024 15:47:03
  2   @      SYSGEN 1   3/18/2024 21:37:31
  2   @      TECO   6   3/18/2024 21:39:35
  2   @      TITLER 1   3/18/2024 21:38:36
  2   @      WAR    3   3/18/2024 21:37:59
  2   IOELEV BIN    8   3/18/2024 21:37:45
  0   ITS    138BIN 8   3/18/2024 21:14:15
  2   ITS    BIN    70   3/18/2024 20:53:08
  1   MACDMP RIM10  1   3/18/2024 21:37:27
  2   MACDMP RIM2   1   3/18/2024 21:37:29
  2   MAGDMP BIN    1   3/18/2024 21:09:45
  0   MAGDMP BIN.KA 1   3/18/2024 22:39:03
  2   SALV   BIN    7   3/18/2024 20:53:27
  0   TS     REDRCT 3   3/18/2024 21:11:47
  2   TV     BIN    16   3/18/2024 21:37:47
*
```

NAMES

- A user name cannot contain more than 6 letters, and are all upper case
- A filename comes in four parts:
  - Device, followed by a colon
  - Folder, followed by a semi-colon
  - First and second parts of the filename, separated by a space
  - Written **Device:Folder;First Second**
- If a device or folder is omitted the current values are used
- The ITS executable has first name (FN1) of @, in folder '.', so full name of '.;@ ITS'

FOLDER LISTING

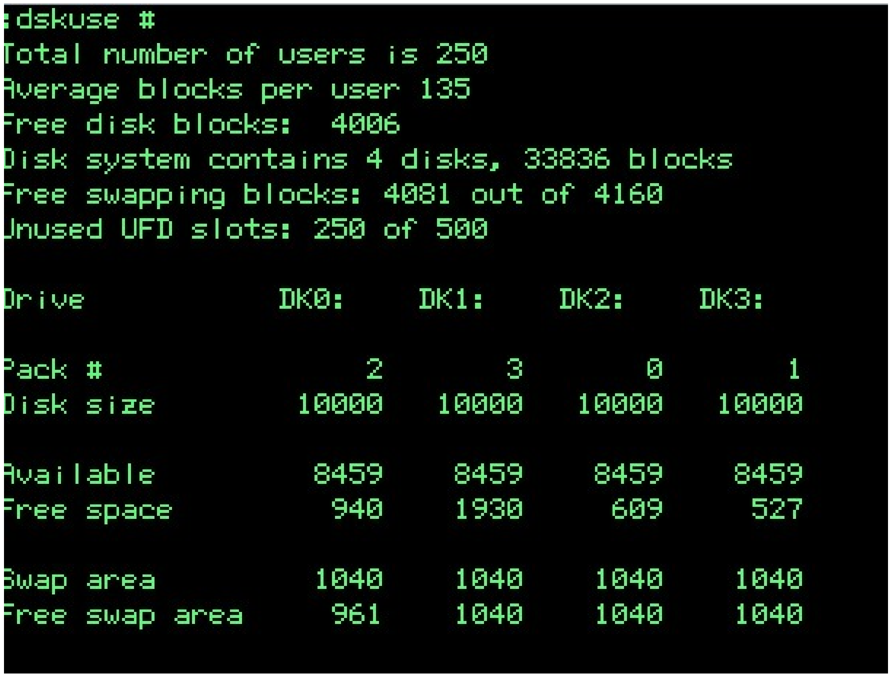
- In the first line is the device (KA) and the folder name (LARS)
- In the second line is the number of free blocks (1K words) for each of the four hard drives, numbered #0 through #3
- Each file is listed in turn:
  - First column is the drive number
  - Second and third columns are the filename
  - Fourth column is file size in 1K word blocks



```
KA    LARS
FREE BLOCKS #2=940 #3=1941 #0=772 #1=527
  3    BACKUP 23      2   11/30/2016 03:56:40
  1    BUBUNV 78      1   11/30/2016 03:56:40
  3    LORENZ 68      1   11/30/2016 03:56:40
  3    MINSKY TRON    1   11/30/2016 03:56:40
  2    MUNCH  340     1   11/30/2016 03:56:40
  3    MUNCH  FASL    1   3/18/2024 21:49:51
  2    MUNCH  LISP    1   11/30/2016 03:56:40
  0    MUNCH  UNFASL  1   3/18/2024 21:49:51
  2    TS     LORENZ  1   3/18/2024 21:38:54
  2    TS     MINSKY  1   3/18/2024 21:38:51
  0    TS     MUNCH   1   3/18/2024 21:38:47
  1    TS     TVBROT  1   3/18/2024 21:38:57
  0    TT2500 57      1   11/30/2016 03:56:40
  1    TVBROT 209     1   11/30/2016 03:56:40
  0    UTNAM  20      1   11/30/2016 03:56:40
```

DISK USAGE

- The disk usage can be found using **:dskuse**
- By itself, **:dskuse** shows the user’s directory, and how much storage has been used
- Using **:dskuse #** shows the disk packs / drives, and how much storage has been used in total
- Using **:dskuse *directory*** shows how much storage has been used in the directory



```
:dskuse #
Total number of users is 250
Average blocks per user 135
Free disk blocks:  4006
Disk system contains 4 disks, 33836 blocks
Free swapping blocks: 4081 out of 4160
Unused UFD slots: 250 of 500

Drive            DK0:     DK1:     DK2:     DK3:

Pack #              2        3        0        1
Disk size       10000    10000    10000    10000

Available        8459     8459     8459     8459
Free space        940     1930      609      527

Swap area        1040     1040     1040     1040
Free swap area    961     1040     1040     1040
```

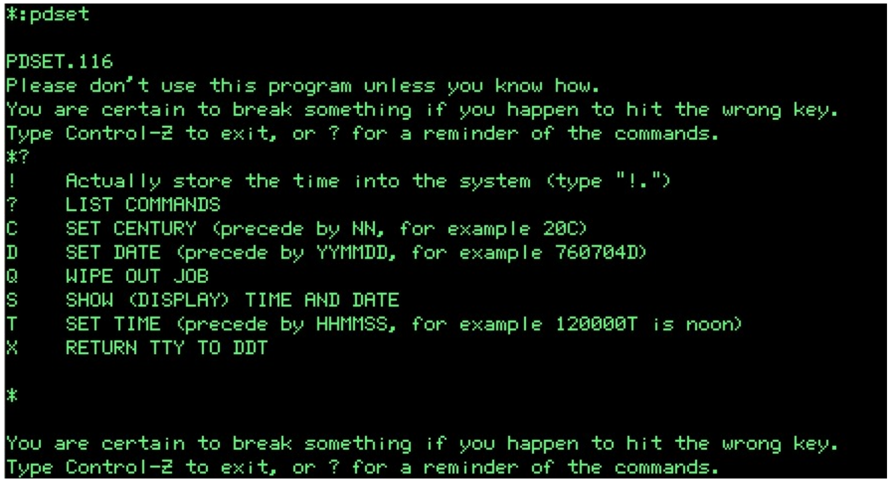
BASIC COMMANDS

- All commands are typed against the left edge of the screen
- Available commands can be listed using **:?**
- Commands are prefixed with a colon, for example to change to another folder use **:CWD *folder***

Command	Shortcut	Description
:LOGIN *user*	*user*<esc>U	Login to account *user*
:LISTF :DIR	Ctrl-F	List current folder
:CWD *folder*		Change current folder to *folder*
:LISTJ	<Esc><Esc> V	List jobs
:CLEAR	Ctrl-L	Clear terminal
:LOGOUT	<Esc><Esc> U	Logout

CHANGING THE SYSTEM DATE AND TIME

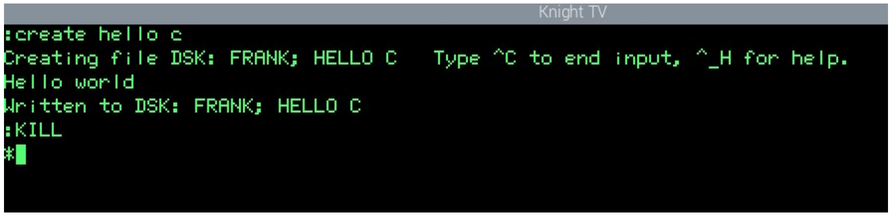
- The system time and date is set using **:PDSET**
- From within the program, you can set the time and date, with commands ***HHMMSST*** and **YYMMDDD** respectively**,** writing it to the system with **!.** and leaving the program with **Q**



```
*:pdset

PDSET.116
Please don't use this program unless you know how.
You are certain to break something if you happen to hit the wrong key.
Type Control-Z to exit, or ? for a reminder of the commands.
*?
!    Actually store the time into the system (type "!.")
?    LIST COMMANDS
C    SET CENTURY (precede by NN, for example 20C)
D    SET DATE (precede by YYMMDD, for example 760704D)
Q    WIPE OUT JOB
S    SHOW (DISPLAY) TIME AND DATE
T    SET TIME (precede by HHMMSS, for example 120000T is noon)
X    RETURN TTY TO DDT

*

You are certain to break something if you happen to hit the wrong key.
Type Control-Z to exit, or ? for a reminder of the commands.
```

CREATING A NEW FILE

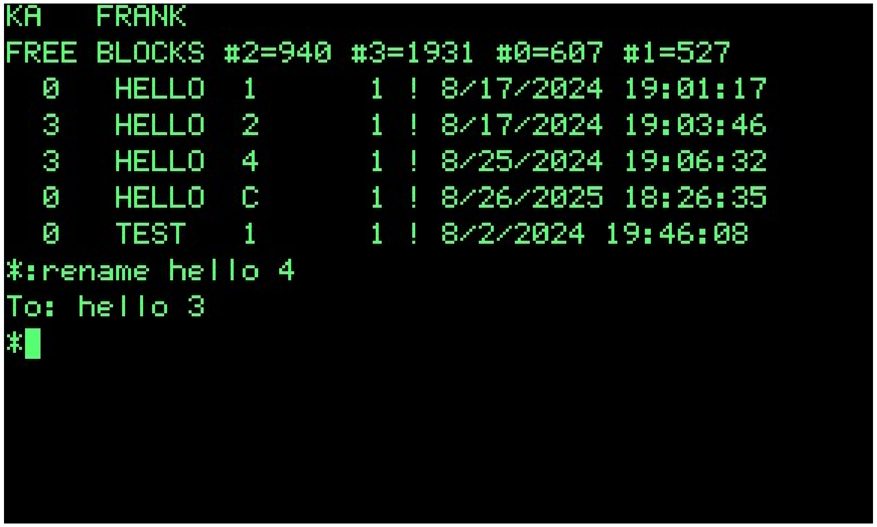
- A new file can be created in the current folder using :**CREATE *filename1 filename2***
- Text can be entered into the file
- When done, press Ctrl-C to save and quit



```
:create hello c
Creating file DSK: FRANK; HELLO C   Type ^C to end input, ^_H for help.
Hello world
Written to DSK: FRANK; HELLO C
:KILL
*
```

COPYING DELETING MOVING / RENAME FILES

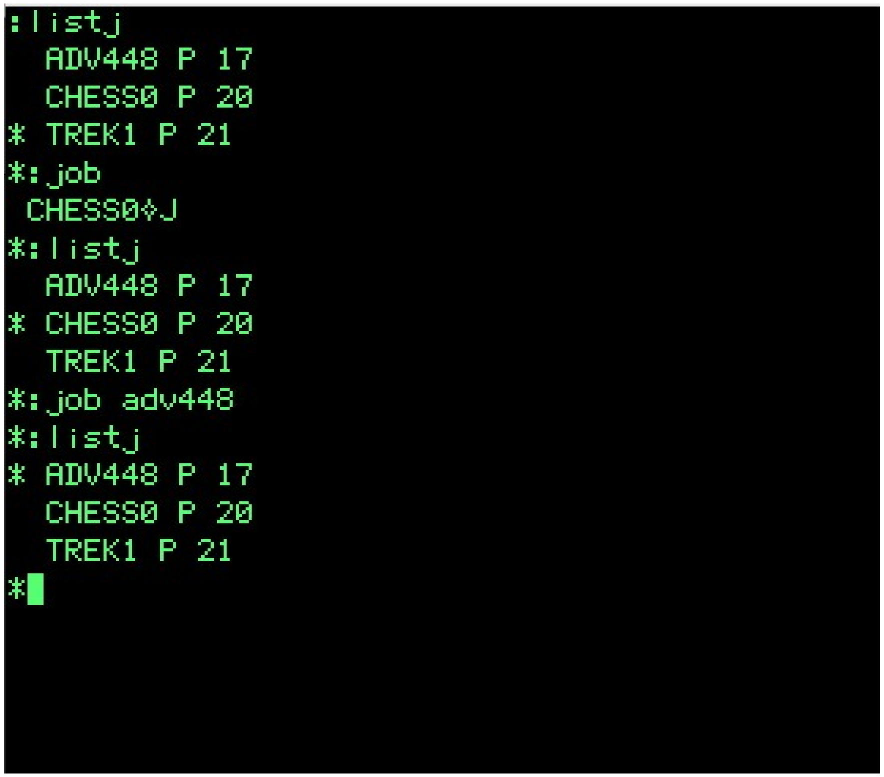
- The command to copy a file is **:COPY filename filename2**
- The command to delete a file is **:DELETE filename filename2**
- The command to move or rename a file is **:RENAME filename filename2**
- Here **filename** and **filename2** are the two six-character parts of the filename
- The system is more interactive than Linux, so for copy and rename, you supply only the original filename, and then ITS will ask for the target filename



```
KA    FRANK
FREE BLOCKS #2=940 #3=1931 #0=607 #1=527
  0   HELLO  1       1 ! 8/17/2024 19:01:17
  3   HELLO  2       1 ! 8/17/2024 19:03:46
  3   HELLO  4       1 ! 8/25/2024 19:06:32
  0   HELLO  C       1 ! 8/26/2025 18:26:35
  0   TEST   1       1 ! 8/2/2024 19:46:08
*:rename hello 4
To: hello 3
*
```

JOBS #1

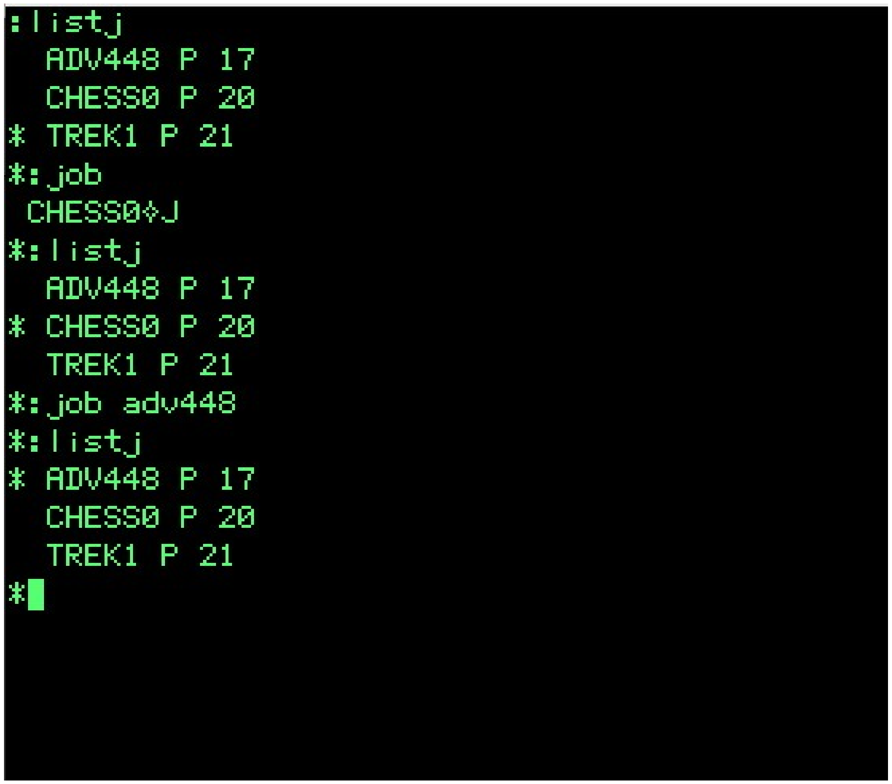
- Multiple jobs can be run simultaneously
- When a process is running, it can be halted by typing Ctrl-Z on a standard terminal, or F1 on a Knight console
- Jobs can then be listed using **:LISTJ**
- The currently select jobs has an asterisk against the name
- The command **:JOB** rotates the selection through the list, or select directly with **:JOB NAME**
- The command **:KILL** kills the currently selected job



```
:listj
  ADV448 P 17
  CHESS0 P 20
* TREK1 P 21
*:job
 CHESS0◊J
*:listj
  ADV448 P 17
* CHESS0 P 20
  TREK1 P 21
*:job adv448
*:listj
* ADV448 P 17
  CHESS0 P 20
  TREK1 P 21
*
```

JOBS #2

- A list of jobs has the following format:
  - First column, asterisk if the job is the currently selected job
  - Second column, the name of the job
  - Third column, the job status
    - R running
    - P stopped
    - W waiting for terminal
  - Fourth column, index number



```
:listj
  ADV448 P 17
  CHESS0 P 20
* TREK1 P 21
*:job
 CHESS0◊J
*:listj
  ADV448 P 17
* CHESS0 P 20
  TREK1 P 21
*:job adv448
*:listj
* ADV448 P 17
  CHESS0 P 20
  TREK1 P 21
*
```

CREATE A NEW USER

https://github.com/larsbrinkhoff/its-manual/wiki/Adding-a-new-ITS-user

- To create a new user, it is necessary to run **^R NAME; ..NEW. (UDIR)**
- The command is entered as shown, except that NAME is replaced by the actual user's name (FRANK)
- A user account is automatically deleted if the folder is empty, and so a new file must be added to the folder, e.g. by running **:CREATE**, and saving the file with **Ctrl-C**
- Run **:HSNAME** to show the current user's name

INTERESTING PROGRAMS

Folder	Command	Description
LARS	MINSKY	Minskytron display
LARS	LORENZ	Lorenz strange attractor simulator
GAMES	CHESS2	Chess game on the Knight terminal
GAMES	TREK	Star Trek game
GAMES	TVWAR	Space War on Knight terminal
SHRDLU	TWDEMO, Ctrl-Q	The SHRDLU AI demonstration on the 340 display

Using TOPS-20

STARTING TOPS-20 ON PIDP10

- When the Raspberry Pi starts, **pdpcontrol start** runs
- The front panel is read to decide which operating system to boot
- With no front panel, in the Raspberry Pi terminal, run **pdpcontrol stop**, wait for the simulation to stop, then run **pdpcontrol start *X***, where *X* is the number of the system to boot
- For TOPS-20 use **pdpcontrol start 200**

Command	Meaning
pdpcontrol stop	Stop PiDP10
pdpcontrol start	Start PiDP10, using the front panel to select the operating system
pdpcontrol start N	Start PiDP10: 1= ITS 2= TOPS-10 200= TOPS-20

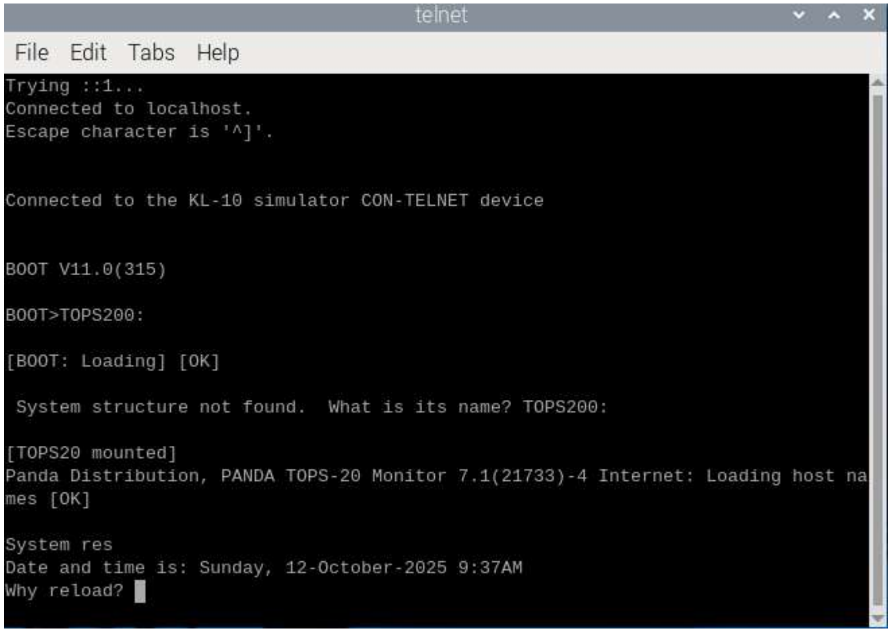
STARTING TERMINAL

- Terminals are started using **pdp *terminal***, or by double clicking on the icon on the left-hand side of the screen
- TOPS-20 doesn’t have a Knight TV console, but it does have the 340 type graphics display

Command	Meaning
pdp con	Starts teleprinter
pdp telcon	Starts telnet
pdp vt52	Starts VT52 terminal

BOOTING THE SYSTEM #1

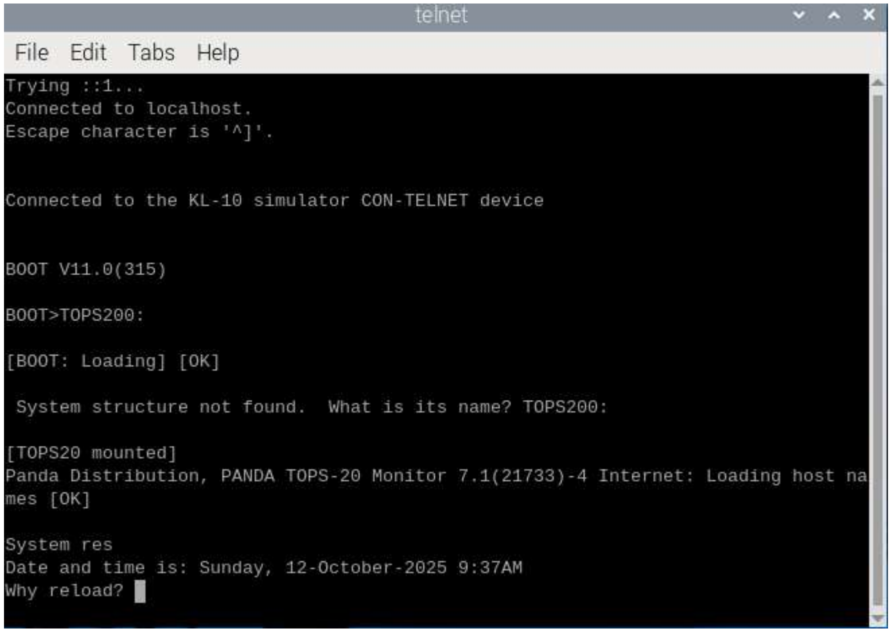
- Use **pdp telcon**, the teletypewriter is too slow during the boot
- (Restarting using **pdpcontrol stop** closes the terminal)
- Wait until the BOOT> prompt appears
- Type **TOPS20:** (include colon)
- When the system complains that it can’t find the correct structure, again type **TOPS20:** (all upper case)



```
Trying ::1...
Connected to localhost.
Escape character is '^]'.


Connected to the KL-10 simulator CON-TELNET device


BOOT V11.0(315)

BOOT>TOPS200:

[BOOT: Loading] [OK]

 System structure not found.  What is its name? TOPS200:

[TOPS20 mounted]
Panda Distribution, PANDA TOPS-20 Monitor 7.1(21733)-4 Internet: Loading host na
mes [OK]

System res
Date and time is: Sunday, 12-October-2025 9:37AM
Why reload? 
```

BOOTING THE SYSTEM #2

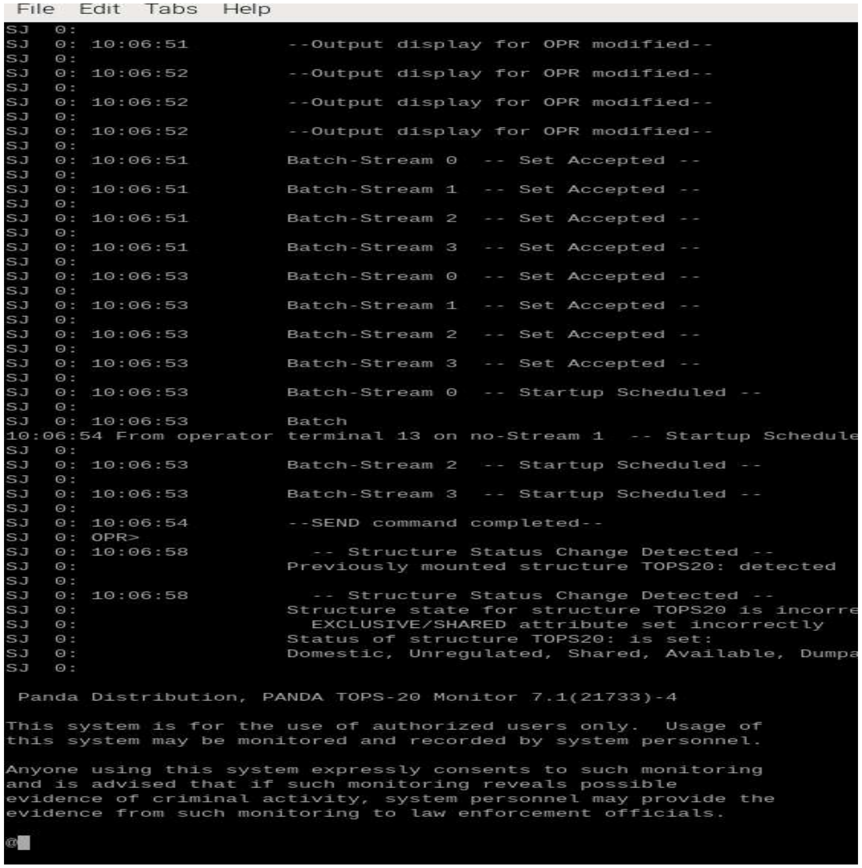
- The system asks why it needs to be reloaded, and you answer **SA**
- The system also asks if it needs to run checks, and the answer is **N**



```
Trying ::1...
Connected to localhost.
Escape character is '^]'.


Connected to the KL-10 simulator CON-TELNET device


BOOT V11.0(315)

BOOT>TOPS200:

[BOOT: Loading] [OK]

 System structure not found.  What is its name? TOPS200:

[TOPS20 mounted]
Panda Distribution, PANDA TOPS-20 Monitor 7.1(21733)-4 Internet: Loading host na
mes [OK]

System res
Date and time is: Sunday, 12-October-2025 9:37AM
Why reload? 
```

BOOTING THE SYSTEM #3

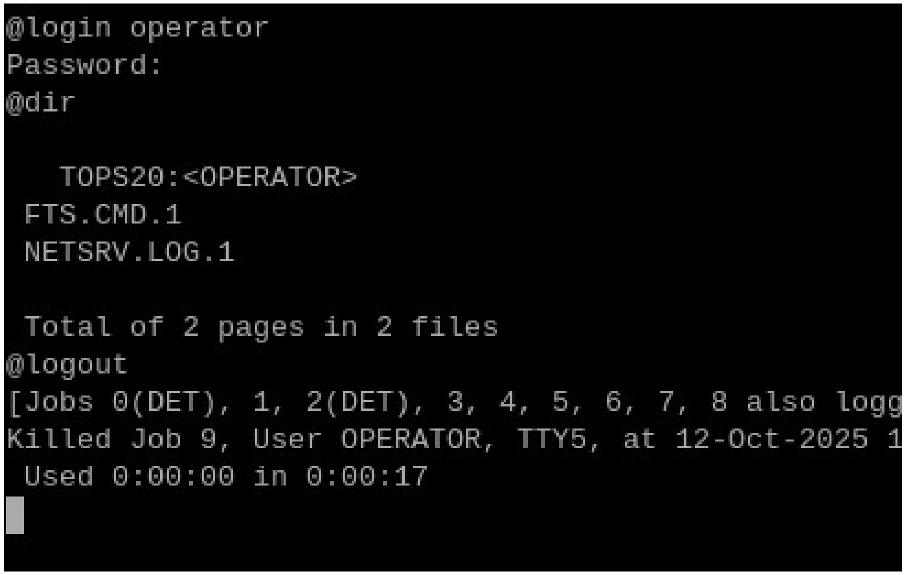
- When the boot text stops being printed, after the sequence starting with SJ, press Ctrl-C and after a couple of paragraphs about Panda the @prompt appears
- The @ is the unprivileged user prompt



```
SJ  0:
SJ  0: 10:06:51        --Output display for OPR modified--
SJ  0:
SJ  0: 10:06:52        --Output display for OPR modified--
SJ  0:
SJ  0: 10:06:52        --Output display for OPR modified--
SJ  0:
SJ  0: 10:06:52        --Output display for OPR modified--
SJ  0:
SJ  0: 10:06:51        Batch-Stream 0  -- Set Accepted --
SJ  0:
SJ  0: 10:06:51        Batch-Stream 1  -- Set Accepted --
SJ  0:
SJ  0: 10:06:51        Batch-Stream 2  -- Set Accepted --
SJ  0:
SJ  0: 10:06:51        Batch-Stream 3  -- Set Accepted --
SJ  0:
SJ  0: 10:06:53        Batch-Stream 0  -- Set Accepted --
SJ  0:
SJ  0: 10:06:53        Batch-Stream 1  -- Set Accepted --
SJ  0:
SJ  0: 10:06:53        Batch-Stream 2  -- Set Accepted --
SJ  0:
SJ  0: 10:06:53        Batch-Stream 3  -- Set Accepted --
SJ  0:
SJ  0: 10:06:53        Batch-Stream 0  -- Startup Scheduled --
SJ  0:
SJ  0: 10:06:53        Batch
10:06:54 From operator terminal 13 on no-Stream 1  -- Startup Schedule
SJ  0:
SJ  0: 10:06:53        Batch-Stream 2  -- Startup Scheduled --
SJ  0:
SJ  0: 10:06:53        Batch-Stream 3  -- Startup Scheduled --
SJ  0:
SJ  0: 10:06:54        --SEND command completed--
SJ  0: OPR>
SJ  0: 10:06:58          -- Structure Status Change Detected --
SJ  0:                 Previously mounted structure TOPS20: detected
SJ  0:
SJ  0: 10:06:58          -- Structure Status Change Detected --
SJ  0:                 Structure state for structure TOPS20 is incorre
SJ  0:                   EXCLUSIVE/SHARED attribute set incorrectly
SJ  0:                 Status of structure TOPS20: is set:
SJ  0:                 Domestic, Unregulated, Shared, Available, Dumpa
SJ  0:

 Panda Distribution, PANDA TOPS-20 Monitor 7.1(21733)-4

This system is for the use of authorized users only.  Usage of
this system may be monitored and recorded by system personnel.

Anyone using this system expressly consents to such monitoring
and is advised that if such monitoring reveals possible
evidence of criminal activity, system personnel may provide the
evidence from such monitoring to law enforcement officials.

@
```

LOGGING IN AND OUT

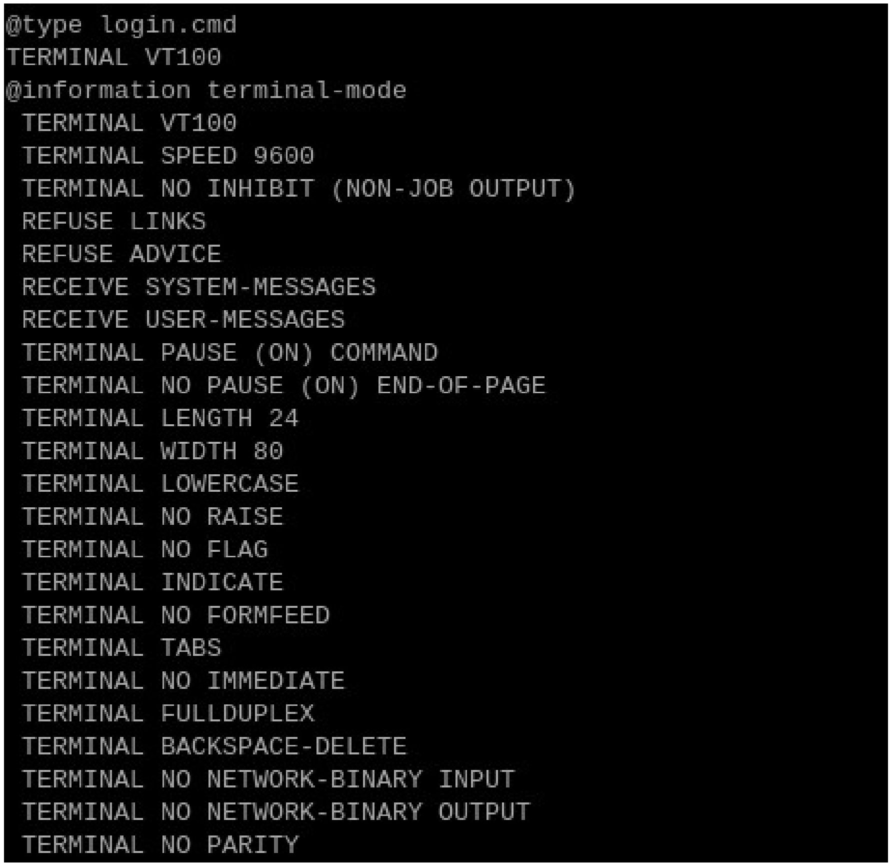
- Type **term vt100** so that the backspace key works properly
- At the @ prompt, type **login operator**
- At the password prompt, type **dec-20**
- Nothing is printed to show that the login is successful, so type **dir**
- Type **logout** to log out



```
@login operator
Password:
@dir

   TOPS20:<OPERATOR>
 FTS.CMD.1
 NETSRV.LOG.1

 Total of 2 pages in 2 files
@logout
[Jobs 0(DET), 1, 2(DET), 3, 4, 5, 6, 7, 8 also logg
Killed Job 9, User OPERATOR, TTY5, at 12-Oct-2025 1
 Used 0:00:00 in 0:00:17
```

RUNNING COMMANDS DURING LOGIN

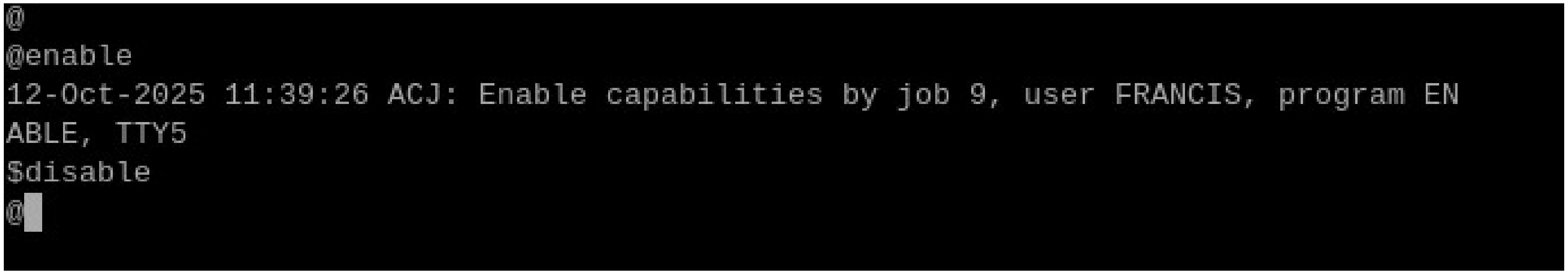
- Create a new file called login.cmd, using at the @ prompt **create login.cmd**
- Into the Emacs screen type **TERMINAL VT100**
- Save the file with Ctrl-X Ctrl-S, and exit Emacs using Ctrl-X Ctrl-C
- The current terminal mode can be displayed using **information terminal-mode**



```
@type login.cmd
TERMINAL VT100
@information terminal-mode
 TERMINAL VT100
 TERMINAL SPEED 9600
 TERMINAL NO INHIBIT (NON-JOB OUTPUT)
 REFUSE LINKS
 REFUSE ADVICE
 RECEIVE SYSTEM-MESSAGES
 RECEIVE USER-MESSAGES
 TERMINAL PAUSE (ON) COMMAND
 TERMINAL NO PAUSE (ON) END-OF-PAGE
 TERMINAL LENGTH 24
 TERMINAL WIDTH 80
 TERMINAL LOWERCASE
 TERMINAL NO RAISE
 TERMINAL NO FLAG
 TERMINAL INDICATE
 TERMINAL NO FORMFEED
 TERMINAL TABS
 TERMINAL NO IMMEDIATE
 TERMINAL FULLDUPLEX
 TERMINAL BACKSPACE-DELETE
 TERMINAL NO NETWORK-BINARY INPUT
 TERMINAL NO NETWORK-BINARY OUTPUT
 TERMINAL NO PARITY
```

PRIVILEGED STATUS

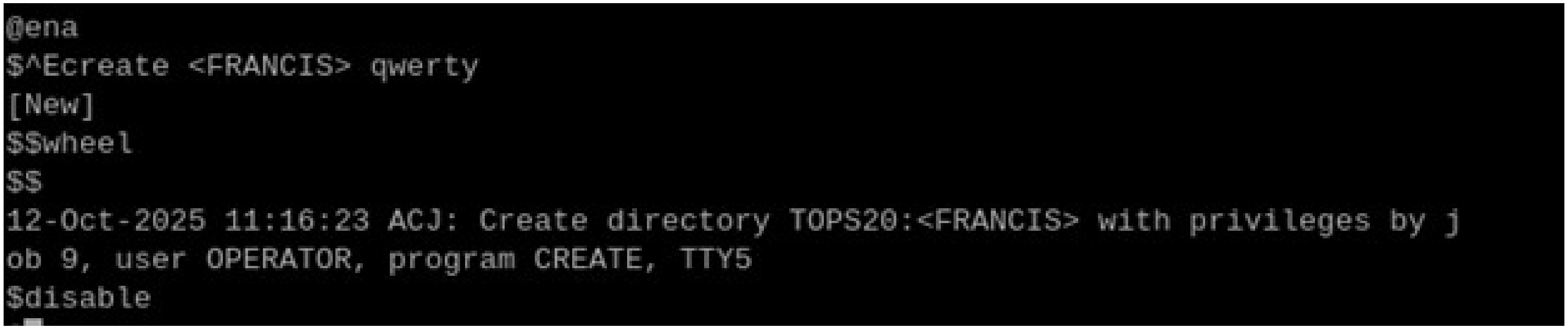
- Privileged status is like sudo or doas on Linux or BSD, requires wheel status
- From the @ prompt, type **enable** or **ena** to enable privileged status, a $ prompt indicates that you have privileged status
- After issuing the commands required at the $ prompt, leave privileged status by typing **disable** or **dis**, which puts you back with the @ prompt



```
@
@enable
12-Oct-2025 11:39:26 ACJ: Enable capabilities by job 9, user FRANCIS, program EN
ABLE, TTY5
$disable
@
```

CREATING A NEW USER #1

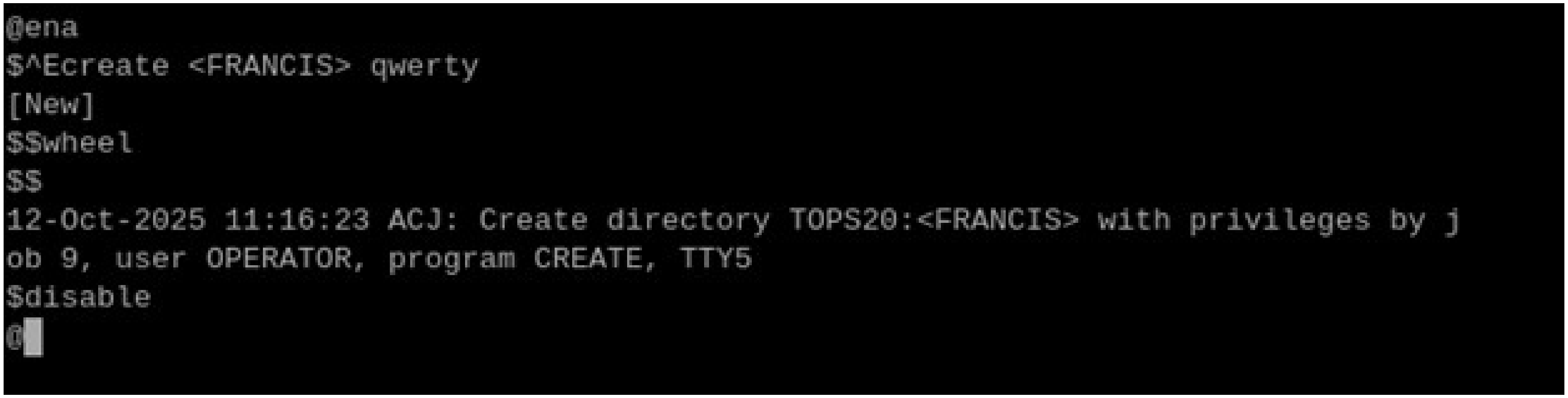
- To create a new user, you must be logged in, and have wheel status
- Enable privileged status at the @ prompt with **enable**
- At the $ prompt, type Ctrl-E**create** ***<username> password***, *for example* Ctrl-E**create <*francis*> qwerty**
- The angled brackets are a mandatory part of the username during creation of the account - the username case is not important



```
@ena
$^Ecreate <FRANCIS> qwerty
[New]
$$wheel
$$
12-Oct-2025 11:16:23 ACJ: Create directory TOPS20:<FRANCIS> with privileges by j
ob 9, user OPERATOR, program CREATE, TTY5
$disable
```

CREATING A NEW USER #2

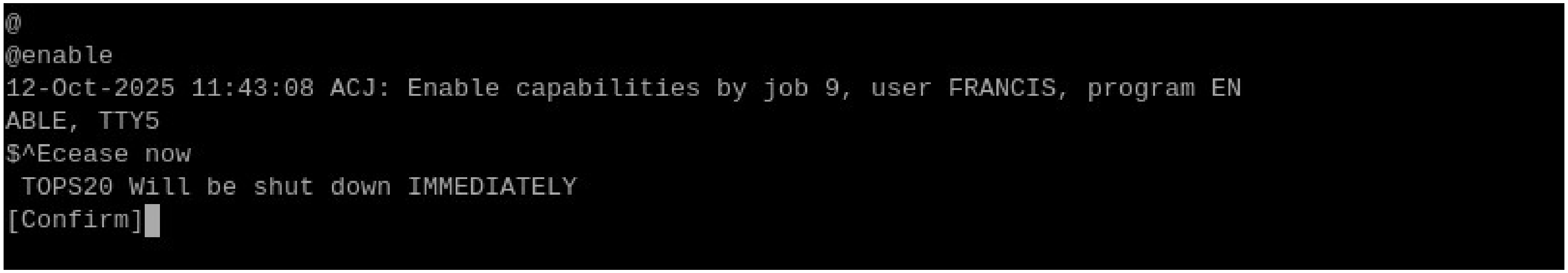
- A new prompt [New] is created by the system, at which point we can enter the user’s privileges
- Against the $$ prompt type **wheel**, and then Ctrl-M to end data entry
- At the $ prompt type **disable** to go back to the normal user status



```
@ena
$^Ecreate <FRANCIS> qwerty
[New]
$$wheel
$$
12-Oct-2025 11:16:23 ACJ: Create directory TOPS20:<FRANCIS> with privileges by j
ob 9, user OPERATOR, program CREATE, TTY5
$disable
@
```

SHUTTING DOWN THE PDP-10

- To shut down the system, you must be logged in, and have wheel status
- Enable privileged status at the @ prompt with **enable**
- Then type Ctrl-E **cease now**
- The system will ask for confirmation, which is supplied by pressing the Enter key
- Wait for the message “Shutdown complete”
- On the Pi terminal, run **pdpcontrol stop**



```
@
@enable
12-Oct-2025 11:43:08 ACJ: Enable capabilities by job 9, user FRANCIS, program EN
ABLE, TTY5
$^Ecease now
 TOPS20 Will be shut down IMMEDIATELY
[Confirm]
```

NAMES

- A user name cannot contain more than 6 letters, and are all upper case
- A filename comes in four parts:
  - Device, followed by a colon
  - Folder, followed by a semi-colon
  - First and second parts of the filename, separated by a space
  - Written **Device:Folder;First Second**
- If a device or folder is omitted the current values are used
- The ITS executable has first name (FN1) of @, in folder '.', so full name of '.;@ ITS'

Emacs

EMACS

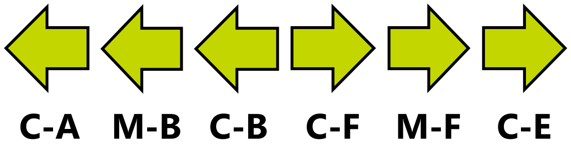
- Emacs can be started with **:EMACS [*filename*]**
- Closing Emacs with Ctrl-X Ctrl-C stops the job, which can be restarted with **:CONTINUE** (or **Esc-P**), it doesn't end Emacs
- Emacs can be ended using **:KILL**

Sequence	Description
Ctrl-X Ctrl-S	Save buffer to file
Ctrl-X Ctrl-C	Close Emacs

EMACS NAVIGATION

https://obsolescenceguaranteed.blogspot.com/2020/02/a-turists-guide-to-its.html

- In the Turists guide to ITS, under "Using the Emacs editor", are shown the keyboard combinations

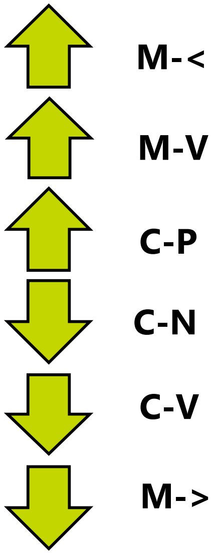


Sequence	Description
Ctrl-F	Forward one char
Ctrl-B	Backwards one char
Meta-F	Forwards one word
Meta-B	Backwards one word
Ctrl-A	Go to start of the line
Ctrl-E	Go to the end of the line

EMACS NAVIGATION

https://obsolescenceguaranteed.blogspot.com/2020/02/a-turists-guide-to-its.html

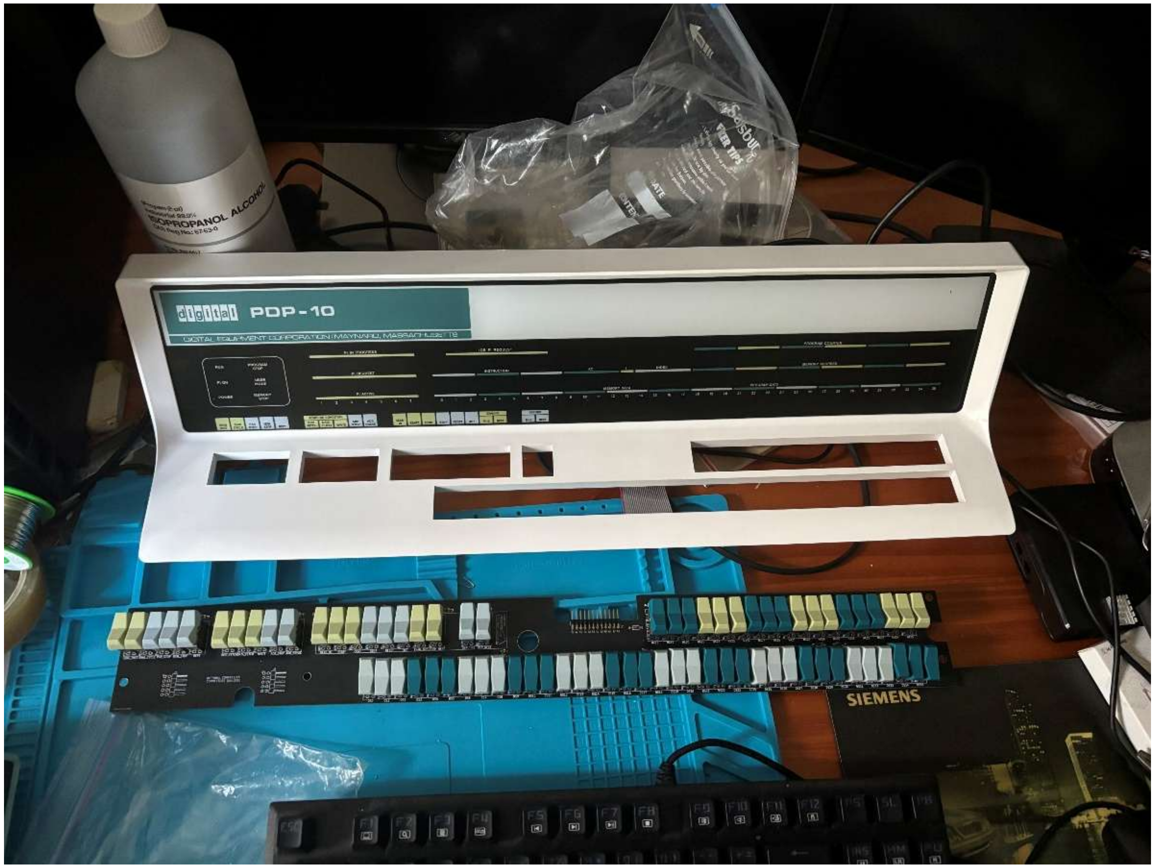
- In the Turists guide to ITS, under “Using the Emacs editor”, are shown the keyboard combinations



Sequence	Description
Ctrl-P	Go to previous line
Ctrl-N	Go to next line
Meta-V	Go up one screen
Ctrl-V	Go down one screen
Meta-<	Go to start
Meta->	Go to end

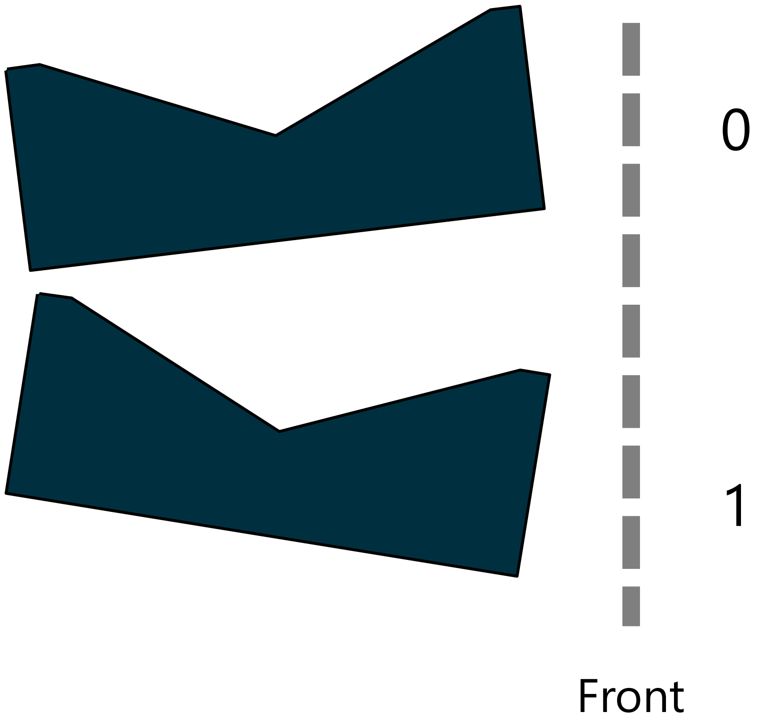
PiDP-10

PIDP-10 OPERATOR CONSOLE



SWITCH POSITIONS

- There are two kinds of switch in the PiDP-10 console:
  - Toggle
  - Momentary
- In both cases, if the switch is higher at the front of the unit than the back, the value is zero
- In both cases, if the switch is higher at the back of the unit than the front, the value is one



BOOTING AN OPERATING SYSTEM #1

- When the Raspberry Pi boots, the rightmost eight data switch positions are used to determine what to do next
- If the switches are set to 001 then the tape-loader for ITS is started, 002 for TOPS 10, 200 for TOPS 20
- The lowest octal digit specifies KA-10, and the higher octal digit specifies KL-10; there is currently no KI simulation
- Any other value, Blinky is run instead

Switches	Operating system
001	ITS
002	TOPS 10
200	TOPS 20

Octal	PDP-10
--X	KA-10
-X-	KI-10
X--	KL-10

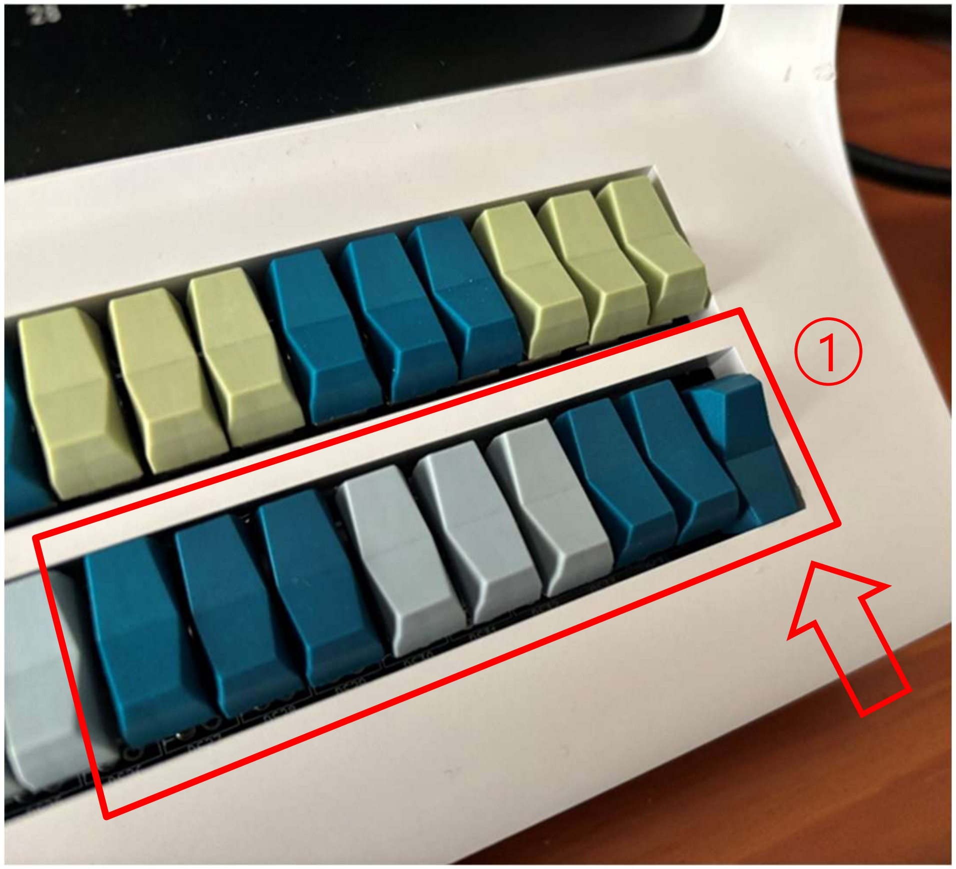
BOOTING AN OPERATING SYSTEM #2

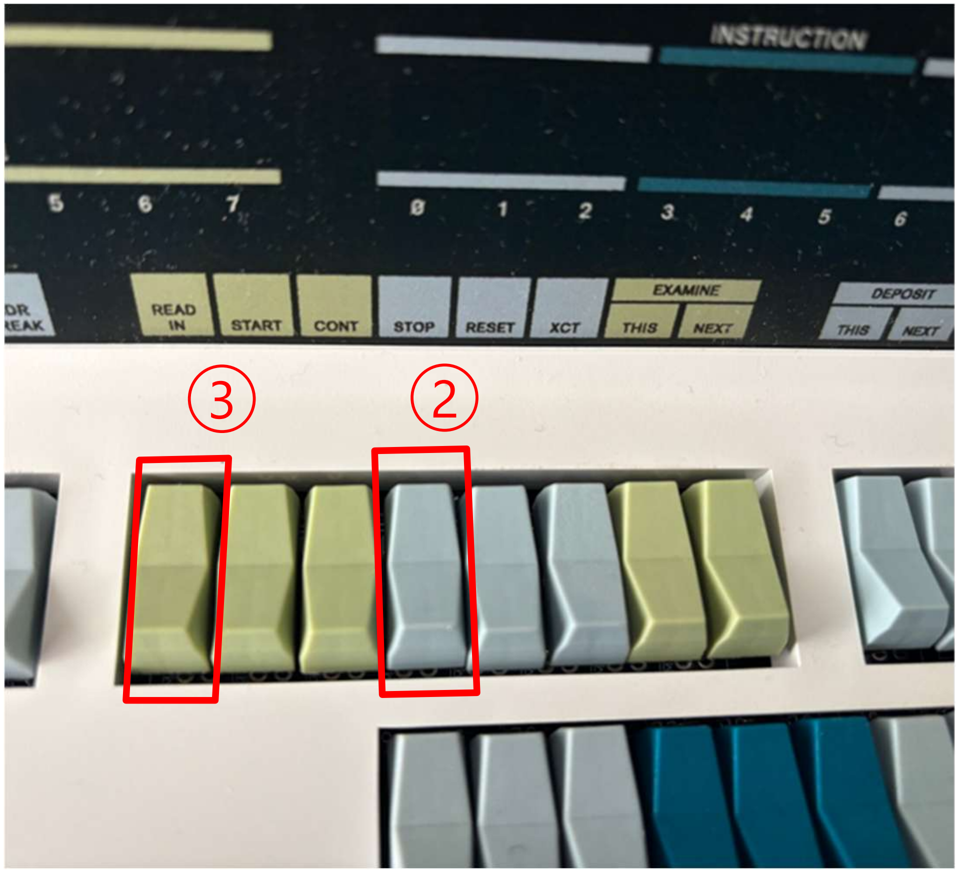
- For ITS, with the paper tape reader, if the PiDP-10 is already powered up, set the switches as before, press momentary switches Stop and then Read In - **pdpcontrol stop**, **pdpcontrol start 1** in effect

Switches	Operating system
001	ITS
002	TOPS 10
200	TOPS 20

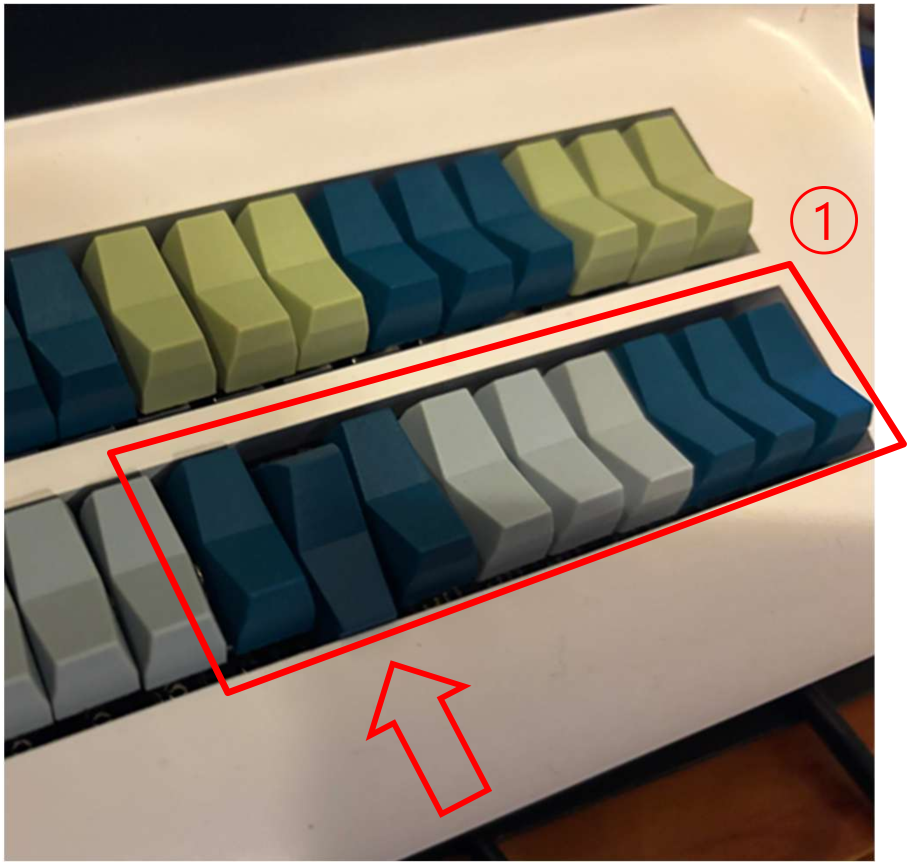
Octal	PDP-10
--X	KA-10
-X-	KI-10
X--	KL-10

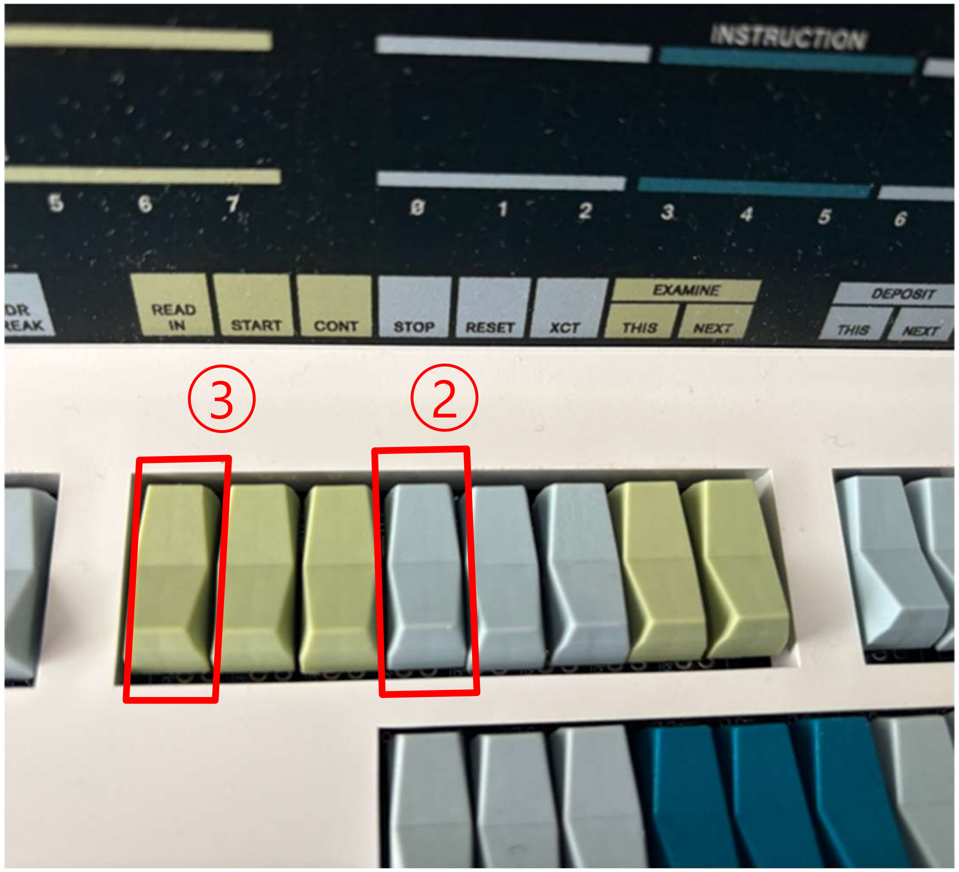
BOOTING ITS


①


INSTRUCTION
5
6
7
0
1
2
3
4
5
6
DR
REAK
READ IN
START
CONT
STOP
RESET
XCT
EXAMINE
THIS
NEXT
DEPOSIT
THIS
NEXT
③
②

BOOTING TOPS-20

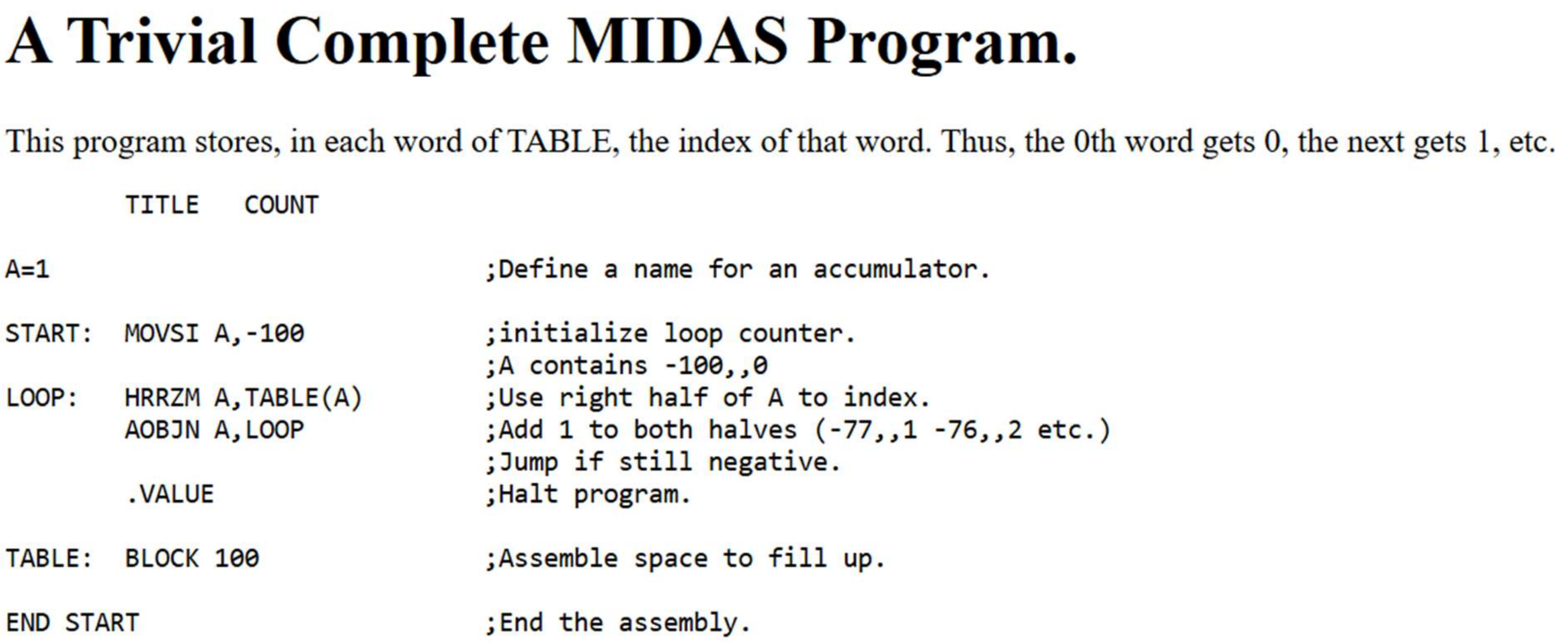




Programming the PDP-10 using MIDAS on ITS

EXAMPLE MIDAS CODE

http://pdp10.nocrew.org/docs/instruction-set/Program.html



A Trivial Complete MIDAS Program.

This program stores, in each word of TABLE, the index of that word. Thus, the 0th word gets 0, the next gets 1, etc.

```
        TITLE   COUNT

A=1                             ;Define a name for an accumulator.

START:  MOVSI A,-100            ;initialize loop counter.
                                ;A contains -100,,0
LOOP:   HRRZM A,TABLE(A)        ;Use right half of A to index.
        AOBJN A,LOOP            ;Add 1 to both halves (-77,,1 -76,,2 etc.)
                                ;Jump if still negative.
        .VALUE                  ;Halt program.

TABLE:  BLOCK 100               ;Assemble space to fill up.

END START                       ;End the assembly.
```

PDP-10 ASSEMBLER

https://bitsavers.org/pdf/dec/pdp10/TOPS10/1973_Assembly_Language_Handbook/01_1973AsmRef_SysRef.pdf

- PDP-10 assembler has two basic formats:
  1. Functions
  2. I/O

Functions

OP	M	A	I	X	Y
0 6	7 8	9 12	13	14 17	18 35

I/O

7	DC	IC	I	X	Y
0 2	3 9	10 12	13	14 17	18 35

OPERATION FIELD OP

- Includes bits 0 .. 6
- Three octal digits (rightmost two bits are field M)
- Functions include:
  - Arithmetic, integer and single precision
  - Logical
  - Shift, rotate, swap register halves
  - Jump, push, pop

MODE FIELD M #1

- The rightmost two bits of the first three octal digits are mode field M (bits 7,8)
- There are four choices - Basic ( ), Immediate (I), Memory (M), and Self/Both (S / B) - which determines the direction flow, from source to destination
- Because these two bits are located within the three octal digits of the function, the handbook lists separate function codes for each value of M

Code	Value	Direction
(none)	0	Memory -> Accum
I	1	Value -> Accum
M	2	Accum -> Memory
S / B	3	Memory -> Memory (and Accum if non zero)

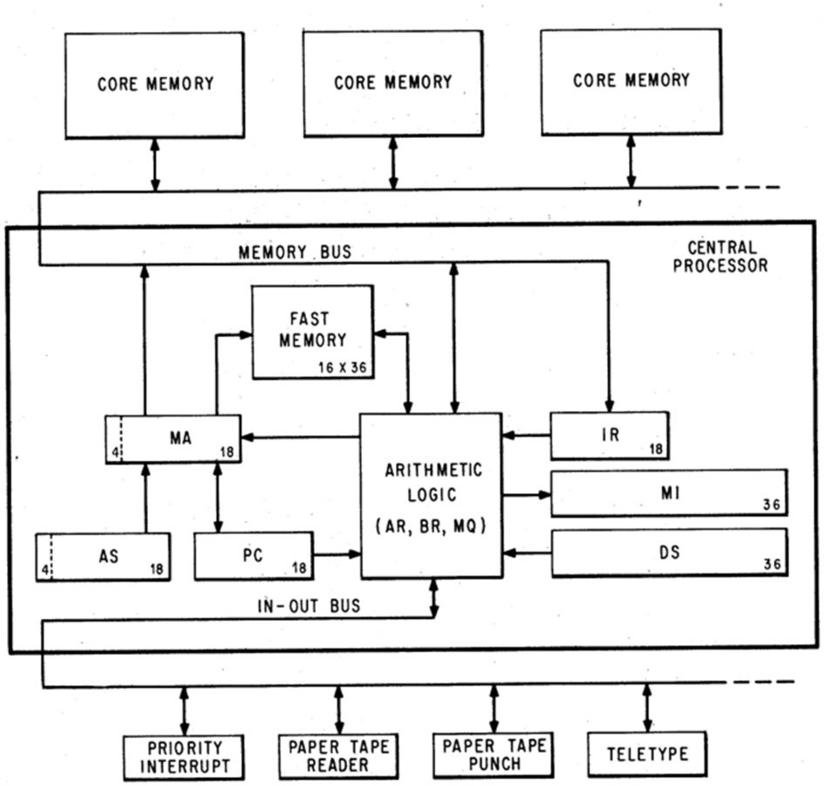
MODE FIELD M #2

- Where the source or destination is Accum then the accumulator is given by the four bits of field A
- Where the source or destination is Memory, the supplied effective address in fields I, X and Y is used to work out where to read or write the data
- If the mode is Self (S) or Both (B) then the accumulator given by field A also gets a copy (if field A is non-zero)

Code	Value	Direction
(none)	0	Memory -> Accum
I	1	Value -> Accum
M	2	Accum -> Memory
S / B	3	Memory -> Memory (and Accum if non zero)

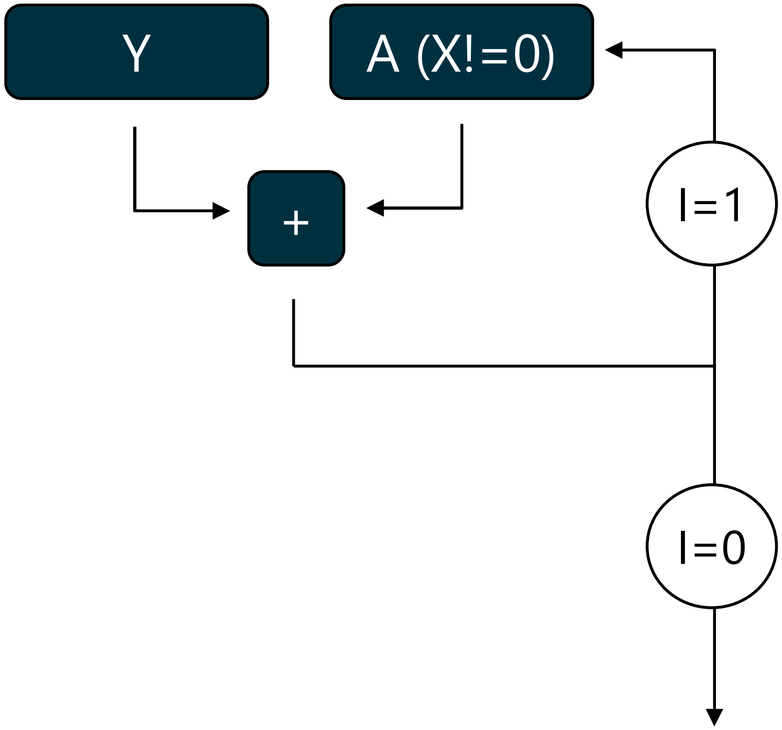
ACCUMULATOR FIELD A

- Where the source or destination is Accum, this is given by the 4 bits of field A, bits 9 through 12
- This references the 16 accumulators in low memory addresses 00_8 through 17_8



EFFECTIVE ADDRESS FIELDS I, X AND Y

- The effective address, E, is constructed from three fields: I, X and Y
- Y is an 18-bit-sized field, from bit 18 through bit 35
- If X is not zero, the contents of the accumulator with number X is added on
- If I is zero, then this is E
- If I is one, the 18 leftmost bits at the location pointed to be E are themselves turned into an effective address, iteratively



I/O INSTRUCTION CODE FIELD IC

- The instruction code is in two parts:
  - Bits 0 - 2, always of value 7_8
  - Bits 10 - 12. which says which command we are doing
- Bits 10 - 12 are within two octal digits
  - Bits 10 and 11 in the first octal digit
  - Bit 12 in the second octal digit
  - Hence the instruction code is 0..3, and then 4 or 0
- With a fixed device ID, the first 13 bits, bits 0 .. 12 are fixed

I/O DEVICE CODE FIELD DC

- The device code is bits 3 through 9, for a total of 7 bits, 128 possible codes
- This is 3 octal digits, with the last digit being 4 or 0
- With a fixed device ID, the first 13 bits, bits 0 .. 12 are fixed
- PTR = Paper Tape Reader

Device	Code	Device	Code
APR	000	DSK	170
PI	004	DTC	320
ADC	024	DTS	324
PTP	100	DLS	240
PTR	104	DC10B/E	240
TTY	120	TMC	340
LPT	124	TMS	344
PLT	140	CCI	014
CR	150		

MOVE [X,,Y]

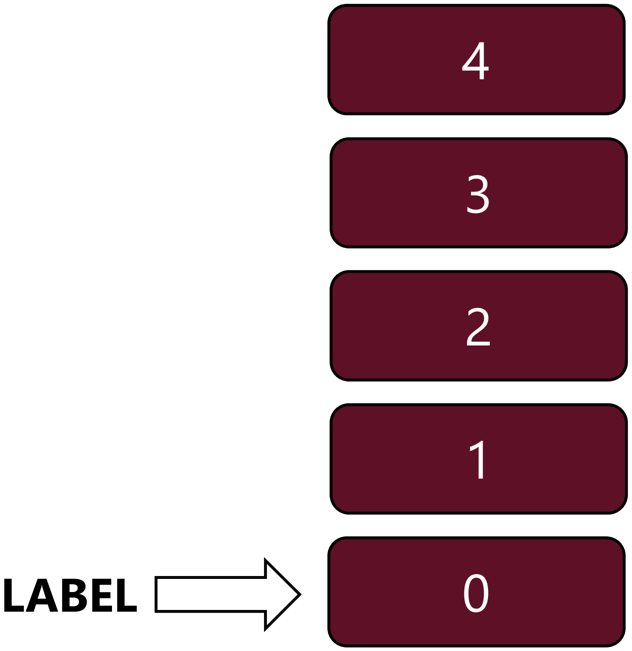
- There are four official ways to use Move, but also there is a fifth way as well provided by the macro assembler
- MOVE can load an immediate half word value using **MOVEI *AC, VALUE***
- MOVE can load from memory using **MOVE *AC, EA***
- The macro assembler can also load an immediate full word, from an address in memory provided by the assembler, of type **MOVE *AC*, [*X,,Y*]**, where X and Y are two halves of a full word of that underlying memory location

Code	Value	Direction
(none)	0	Memory -> Accum
(none)	0	[X,,Y] -> Accum
I	1	Value -> Accum
M	2	Accum -> Memory
S / B	3	Memory -> Memory (and Accum if non zero)

BLOCKS

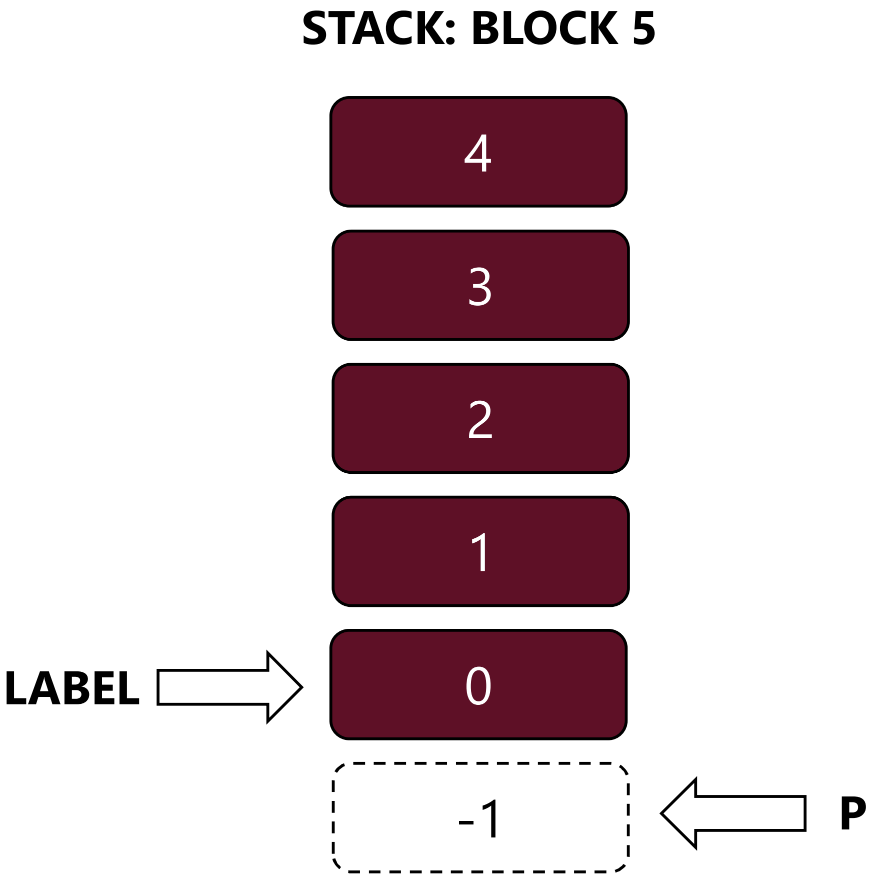
- A block is defined using **BLOCK**
- The format is ***LABEL*:   BLOCK *N***
- This reserves N words of memory, between *LABEL* and *LABEL*+(*N*-1), which can be accessed as a table using the *LABEL* address
- The block is usually defined at the end of the code, after the code which terminates the program

LABEL: BLOCK 5



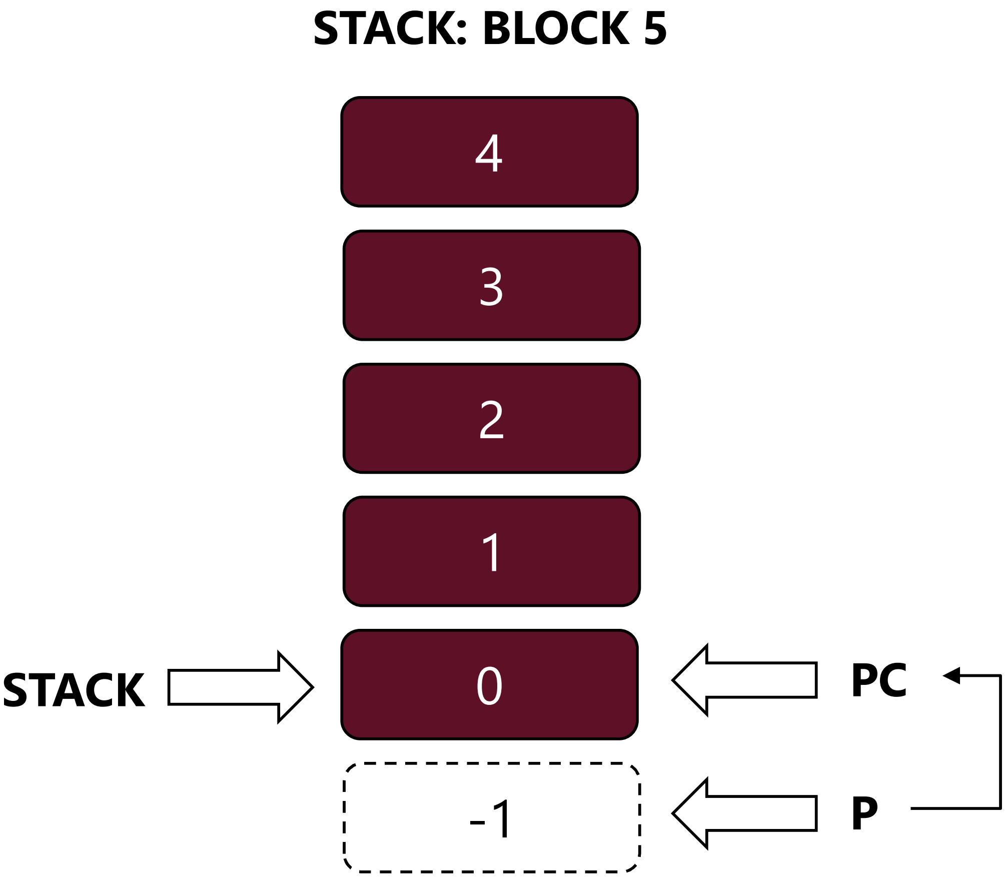
STACK AND STACK POINTER #1

- The PDP-10 doesn’t have a dedicated stack pointer
- Instead, one of the accumulators, by convention $AC17_8$ is used as the stack pointer, and the stack pointer variable is conventionally called P
- The name P is defined to be 17_8, using **P=17** at the start of the code
- Any accumulator can be used as a stack pointer, but AC17 is typically used as it's an index register and out of the way of other uses for accumulators.



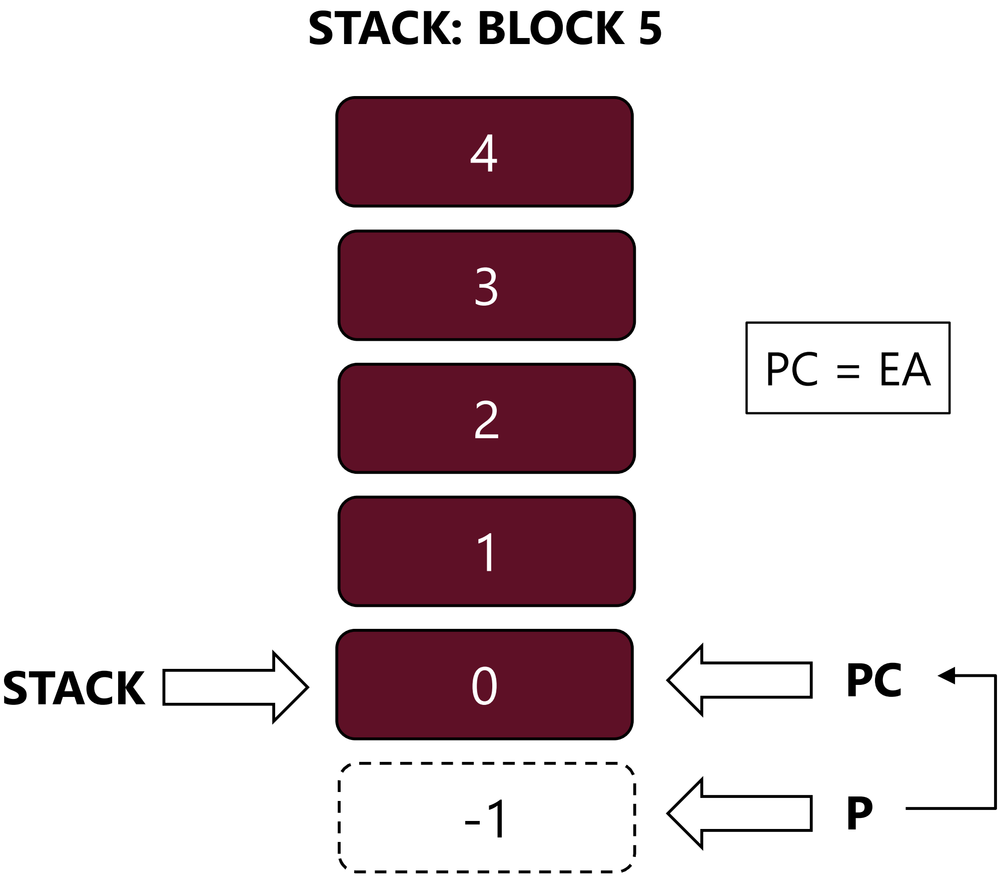
STACK AND STACK POINTER #2

- A table is created using **STACK: BLOCK *N***, and enabled using **MOVE P, [-N,,STACK-1]**
  - The upper 18 bits holds the count into the stack; the bottom 18 bits holds an address just below the stack
  - It is STACK-1, because when pushing data, the address is incremented first before storing the value
  - It is -N because the counter is incremented as well; if it becomes zero then an overflow occurs



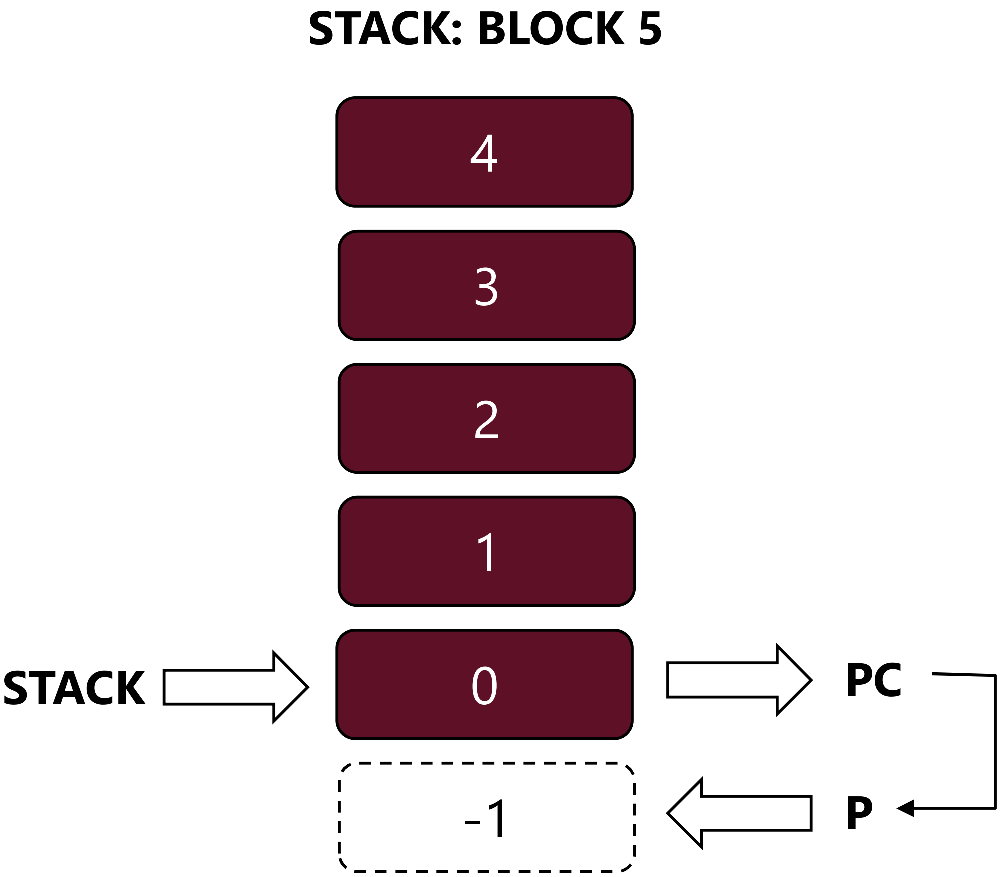
CALL SUBROUTINE #1

- With the stack pointer and stack defined, a subroutine can be called using **PUSHJ *P, EA***
- It does the following actions:
    - Increment the stack pointer and stack counter
    - Put the current program counter in the stack table at stack pointer
    - Put the new effective address into the program counter



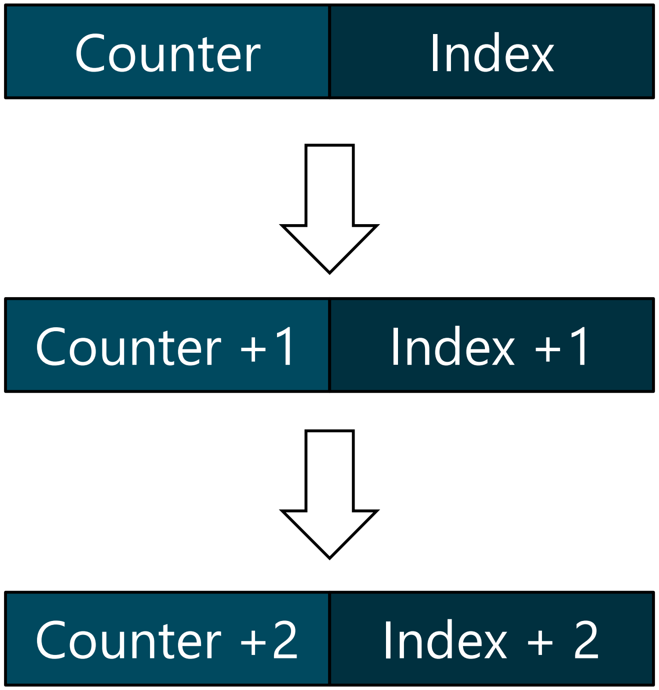
CALL SUBROUTINE #2

- At the end of the subroutine, **POPJ P,** does the reverse of PUSHJ
- The following actions are done:
  - The value at the top of the stack is put into the program counter;
  - then the count in the accumulator is decremented, as is the stack pointer



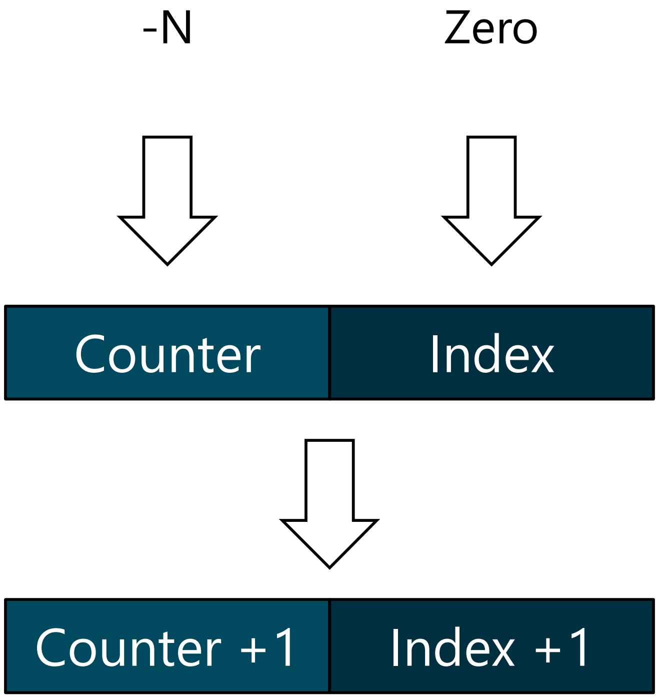
FOR LOOP #1

- A *For* .. *Next* loop requires two things, a count and a start of the loop
- The start of the loop is given by a label, or a relative address to the current address
- The count is held in an accumulator. The left-hand part contains the counter; the right-hand part contains an index.
- Each time around the loop, both the counter and index are incremented, and we keep doing this until counter is zero



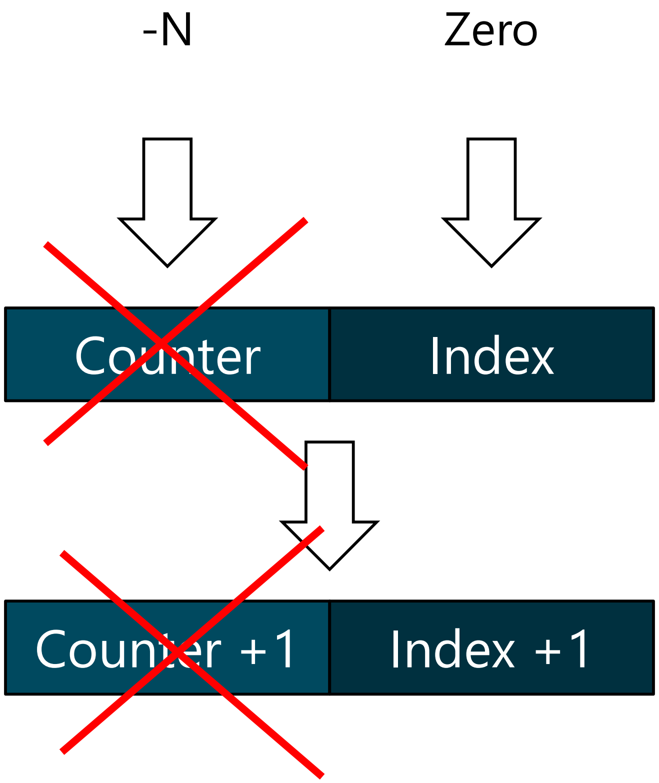
FOR LOOP #2

- The accumulator can be set up with **MOVSI *AC, -N*** for N loops
- This instruction moves the value of -N into the left-hand, counter, side of the accumulator (rather the right-hand, index, side with **MOVEI *AC, N***) and sets the index to zero
- Then at the end of the loop, we use **AOBJN *AC, LABEL***
- We add 1 to each half of the accumulator, and jump back to LABEL if stick negative

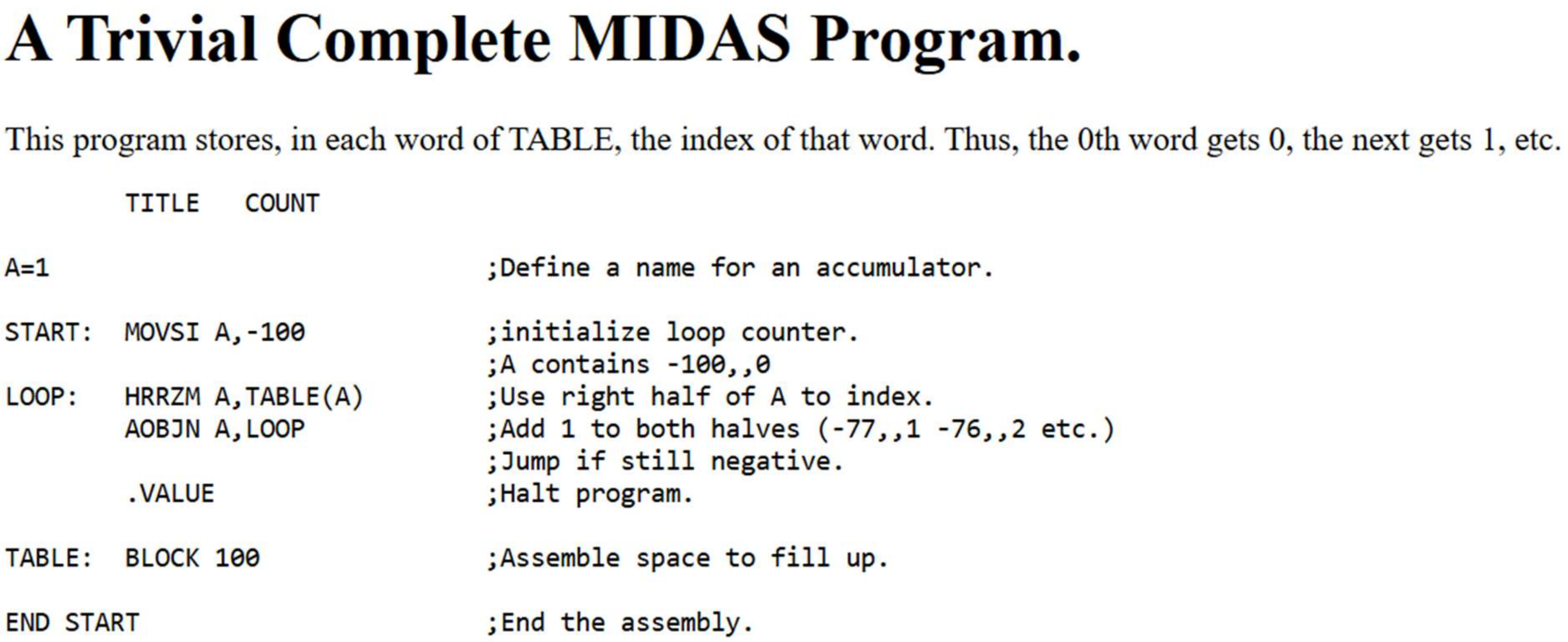


FOR LOOP #3

- The index can be used to step through an array
- Once a block has been created with a *LABEL*, we can access it using the counter / index pair
- When an effective address is computed, only the bottom 18 bits of the index is used, the counter part is not used
- The block is indexed in the form *LABEL*(*AC*), where AC is the accumulator containing the counter / index pair and providing the index of the EA



FOR LOOP #4



A Trivial Complete MIDAS Program.

This program stores, in each word of TABLE, the index of that word. Thus, the 0th word gets 0, the next gets 1, etc.

```
        TITLE   COUNT

A=1                             ;Define a name for an accumulator.

START:  MOVSI A,-100            ;initialize loop counter.
                                ;A contains -100,,0
LOOP:   HRRZM A,TABLE(A)        ;Use right half of A to index.
        AOBJN A,LOOP            ;Add 1 to both halves (-77,,1 -76,,2 etc.)
                                ;Jump if still negative.
        .VALUE                  ;Halt program.

TABLE:  BLOCK 100               ;Assemble space to fill up.

END START                       ;End the assembly.
```