The Unichannel-15

The Unichannel-15 (UC15) was a PDP-11/05 system that attached to a PDP-15 system and acted as an I/O processor. It was created by the PDP-15 group as a way of accessing inexpensive Unibus peripherals, like the LP11 and RK11, without having to do unique controllers. The intention was to support new peripherals as they came out, but the next wave of PDP-11 storage devices (the RL11 and RK611) did not support 18b data. By the time new 18b-compliant storage devices became available (the RH11C/Massbus disks), the PDP-15 was essentially gone.

The UC15 consisted of four distinct pieces:

- A PDP-11/05 system with 4KW-12KW of memory. (In theory, other Unibus CPUs could be used instead, but according to Barry Rubinson, chief designer of PIREX, only 11/05s were shipped.)
- A memory access port that cross-connected part of the PDP-11/05's Unibus address space to the PDP-15's memory.
- A DR15-C custom parallel interface that talked to two DR11-C standard parallel interfaces in the PDP-11/05.
- Various PDP-11 peripherals. The clock and console port were standard and were private to the PDP-11/05, but all the others were accessible to the PDP-15 (via the PDP-11 PIREX software).

The UC15 is not well documented, but fortunately the schematics are on line. This allows for accurate reconstruction of the functionality.

<u>Memory Model</u>

The memory model was rather complicated. On the PDP-11, the lowest part of its address space was occupied by local 16b memory, called "private" memory. From the end of private memory to the end of Unibus address space (124KW on a PDP-11/05), the address space was mapped into the bottom of PDP-15 memory. Above that was the IO page. Because the PDP-11/05 could only address 28KW, only some of the PDP-15's memory could be accessed by the PDP-11 processor; the rest was accessible only by Unibus DMA devices.

The PDP-15's memory model also had multiple segments. The lowest part was accessible to the PDP-11/05 processor. The next segment was accessible from Unibus DMA devices but not from the PDP-11 CPU. The highest segment was private.

Graphically (all addresses in words):

```
                          +----------+ 128K
                          |  private |
                          |  memory  |
   128K +----------+     /+----------+ 120K-108K
        |    IO    |   / |accessible|
        |   page   |  /  |  memory  |
   124K +----------+/   /+----------+ 24K-16K
        |  Unibus  |   / |  shared  |
        |   space  |  /  |  memory  |
    28K +----------+/   /+----------+ 0
        |  shared  |   /    PDP-15
        |  memory  |  /
 4K-12K +----------+/
        |  private |
        |  memory  |
      0 +----------+
           PDP-11
```

When the PDP-11 processor accessed shared memory, it generated 16b references. On reads, the top two bits of PDP-15 memory were ignored. On writes, only bits <2:9>, <10:17>, or <2:17> of the PDP-15 memory word could be updated; the top two bits were always cleared. However, when DMA devices accessed shared or accessible memory, they could (potentially) read or write all 18b, using the Unibus parity lines as extra data lines. The RK15 disk controller, in particular, absorbed and delivered 18b of data, always using word accesses. All other supported peripherals used programmed I/O and could not access PDP-15 memory at all.

Interprocessor Communication

The PDP-15 used a master-slave model to talk to the UC15 and its operating program (PIREX). To start an operation, the PDP-15 wrote the address of a Task Control Block (TCB) into the TCB pointer register. Although this register was 18b wide, in practice TCBs had to be in shared memory, limiting TCB addresses to 15b.

Writing the TCB pointer set a flag in one of the PDP-11's DR11-Cs, potentially causing an interrupt. PIREX then copied over the TCB and initiated the requested operation. When the operation was done, PIREX posted a completion notice back to the PDP-15 using one of four dedicated API interrupts. The PDP-15 could then initiate another operation to that device. The availability of multiple completion interrupts allowed the PDP-15 to initiate multiple concurrent IO requests, up to the limit of PIREX's queue storage.

On the PDP-15 side, the interprocessor communications interface was called the DR15-C, a custom parallel interface. It had a 1-bit persistent interrupt enable, a write-only 18b Task Control Block pointer, four flags wired to API interrupts on levels 0-3, and an "IO access flag" that indicated whether the PDP-11 had read the TCB pointer register. It was programmed via devices 60 and 61:

SIOA	706001	skip if the IO access flag is set
CIOD	706002	clear the IO access flag
(LIORS)	706004	write AC<1:2> to 767774<1:0> write AC<3:17> to 767764<15:1> set 767760<7>, with possible interrupt
LIOR	706006	CIOD + LIORS
CAPIn	7061(n*2)1	clear APIn flag
(ODRS)	706102	OR DR-15 status to AC
RDRS	706112	clear AC, then OR DR15-C status to AC bit<17> = interrupt enable interrupt enable is SET on reset or CAF
LDRS	706122	load DR15-C status from AC bit<17> = interrupt enable
SAPIn	7061(n*2)4	skip if APIn flag is set

A couple of fine points. First, the IO access flag is not wired to interrupt, either PI or API. That means the PDP-15 must poll to see if the PDP-11 is ready to accept a new command. If the PDP-11 is hung or has crashed, the PDP-15 may hang is well. Third, the API vectors are actually stored in the DR11-C output buffers; the DRC15-C does not implement local buffering. Third, the interrupt enable acts as an AND on assertion of either PI or API interrupts. If APIn flag sets while interrupts are disabled, no interrupt is requested; but an interrupt will be requested when interrupt enable is set. In short, DR15-C interrupts are in effect level sensitive, not edge-sensitive.

On the PDP-11 side, two standard DR11-Cs are used.

DR11-C #0 - BR5, vector 300

767770	<7> = API done: logical AND of 767774<15:14,7:6> <6> = interrupt enable
767772	byte write to initiate an API0 interrupt <6:0> = API vector (0-177)
767773	byte write to initiate an API1 interrupt <6:0> = API vector (0-177)

```
777774    <1:0> = bits <1:2> of TCB pointer
          <6> = API2 done (request not asserted)
          <7> = API0 done
          <11:8> = size of PDP-11 local memory in 4KW chunks
          <14> = API3 done
          <15> = API1 done

DR11-C #1 - BR7, vector 310

767760    <7> = new TCB pointer flag
                set by PDP-15 LIOR(S)
                cleared by DATI to 767764
          <6> = interrupt enable
767762    byte write to initiate an API2 interrupt
          <6:0> = API vector (0-177)

767763    byte write to initiate an API3 interrupt
          <6:0> = API vector (0-177)

767774    TCB pointer<3:17>; bit<0> is always 0
```

Although the PDP-11 has the ability to be interrupted when the PDP-15 has processed all outstanding API interrupts, this capability is not used in PIREX. Indeed, PIREX doesn't allocate a vector at 300 or ever access program register 767770.

Peripherals

The PDP-11/05 included a KW11-L compatible line time clock and a KL11 compatible serial interface as standard equipment. The UC15 added two DR11-Cs and some number of Unibus peripherals. The following were supported:

RK15	18b version of the RK11
LP11	line printer
LS11	serial line printer, program compatible with the LP11
LV11	electrostatic printer/plotter, program compatible with the LP11
XY11	plotter
XY311	plotter, program compatible with the XY11
CR11	card reader

In addition, the UC15 could support the so-called "Hurley protocol" serial network interface to a DECsystem-10 over a serial line, using either the console serial port or a DL11, plus a KG11, at their default IO space addresses and vectors. Note that the default DL11 vector was 300, conflicting with DR11C #0's interrupt, which PIREX did not use.

There was unsupported code for the PDP-11 EAE and the TC11 DECtape controller. The TC11 could not read or write 18b data in DMA mode, and there was already a DECtape controller native to the PDP-15.

History

It’s difficult at this date to ascertain how successful the UC15 was as a product. The online listing of 18b processors stops at 1972, just before the UC15 was introduced; at that point, more than half the PDP-15s produced had already been shipped. *Computer Engineering*, the semi-official history of DEC’s systems, praises the ingenuity of the engineering effort but has nothing to say about the commercial results. Mike Ross’s XVM systems both had UC15’s, although one of them was partially dismantled and the PDP-11 removed. Max Burnet reports that the PDP-15 was not popular in Australia, and none of the eleven systems sold there had a Unichannel.

Acknowledgements

As usual, Al Kossow’s Bitsavers archive played a critical role in understanding the UC15. It contains both the schematics for the components and the sources for XVM/DOS, including PIREX and other bits and pieces of Unichannel software. Max Burnet and Barry Rubinson provided anecdotal information on the history of the system.