Pac-Man for the DEC VT420

François Laagel, IEEE

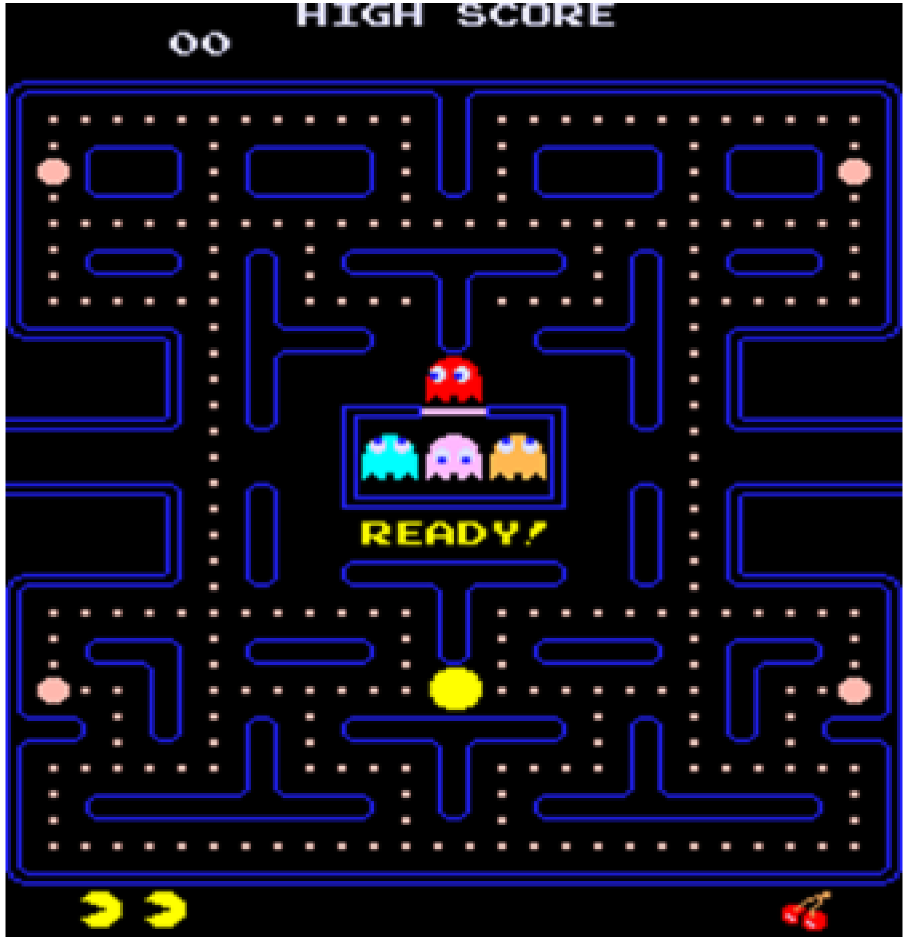
Introduction to Z79Forth

- Hardware (HD6309 based SBC) and software platform for learning Forth and hardware experimentation.
- HW design started at the end of 2018 and lasted about 3 months.
- Forth is the OS and mostly EEPROM resident.
- Code originally written for the Z80 in 1983 (EDTASM+ by Microsoft).
- Either a 79-STANDARD sub-set or an extended ANS94 core word set.
- 32 KB SRAM, 8 KB EEPROM, mass storage over CompactFlash (64 MB).
- Asynchronous serial line for the console (USB or RS232 based).
- USB powered.

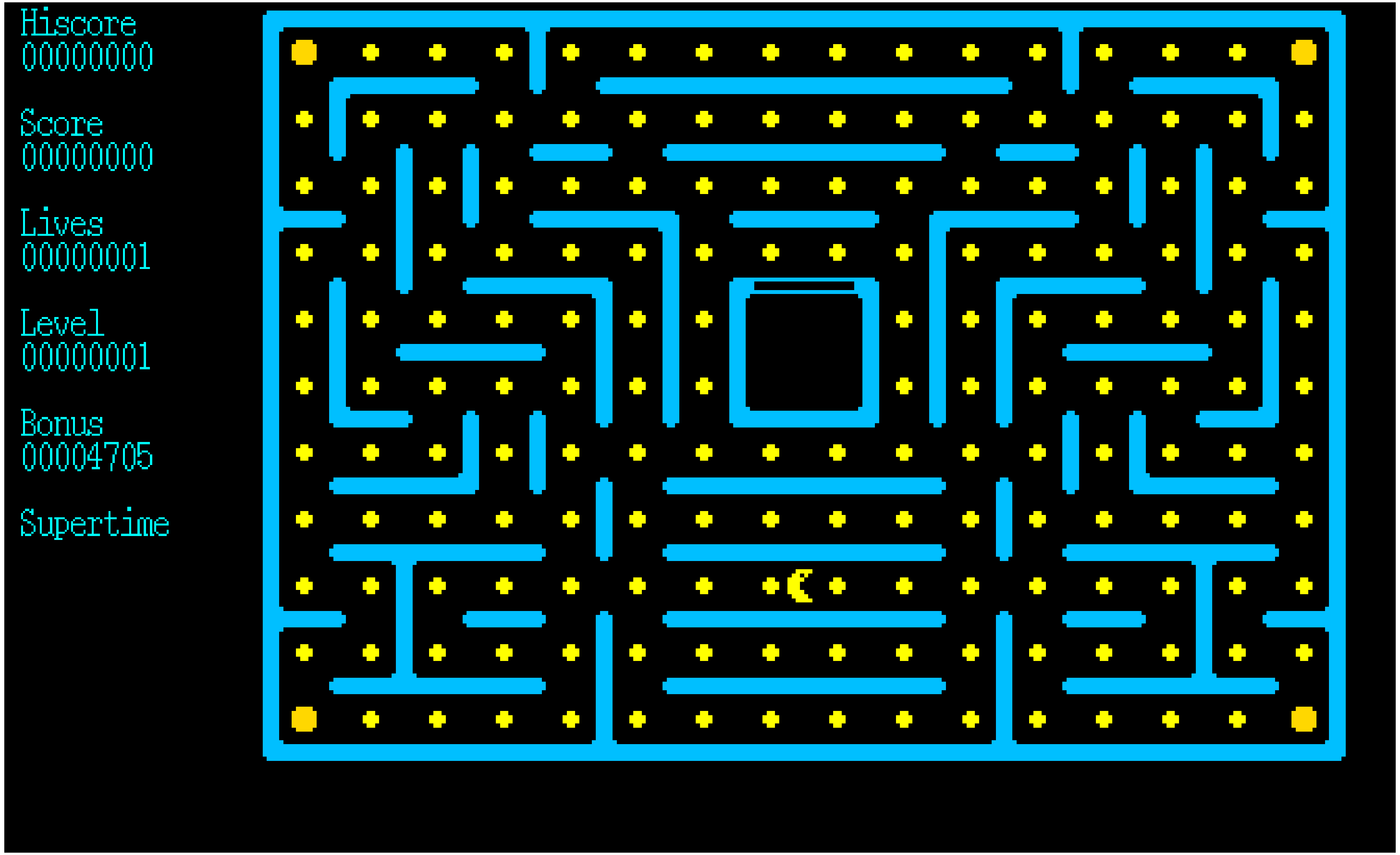
The DEC VT420

- *Low-cost* (by DEC standards) monochrome text based terminal.
- Intel 8031 based design, introduced in 1990.
- Supports two simultaneous sessions (EIA 232 and DEC 423).
- A 14 inch display supporting 800 x 400 resolution.
- ANSI (VT52 mode) compliant control sequences.
- Software flow controlled (XON/XOFF) serial communications.
- Supports user defined downloadable fonts (`DECDLD`).

Pac-Man Fundamentals - The NAMCO Maze



The Thornaes Maze. X11/POSIX, C++, 1995



Software Development Methodology

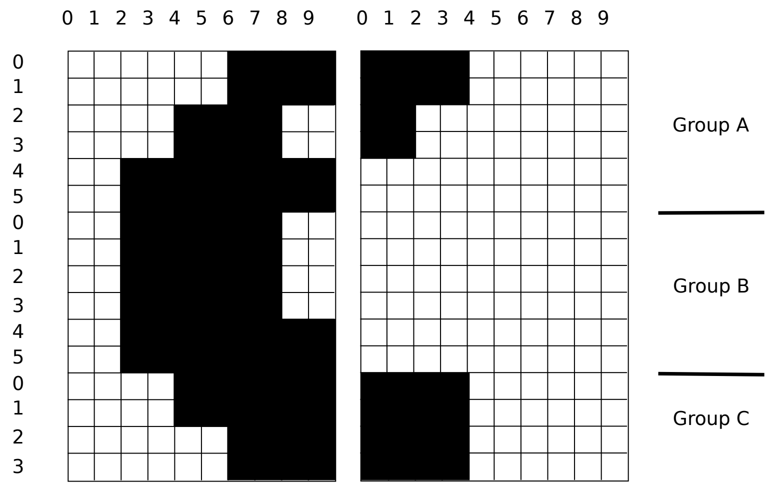
- **Presentation Layer** Design of a user-defined font for the VT420.
- **Basic Business Layer** Object identification and definition.
  - Shared ghost moving policy (initially random). Preserving erasables.
  - Specific Pac-Man moving policy. Gathering collectibles, score maintenance, collision detection, remaining lives bookkeeping.
- **Successive Business Layer Refinements** Ghost mode management, differentiated moving policies (aka. *AI Integration*).
- **Testing**
- **Specific Support for the VT340**
- **Publish the Forth Code**
- **Linux/C Port**
- **OpenVMS/C Port**

Presentation Layer - Part I: Design Choices

- Select an object oriented approach.
- Select Thornaes maze topology.
- In 80x24 mode, the VT420 fits Thornaes's design nicely if we view it as 33 double width characters horizontally by 23 physical rows.
- Custom 30 double width character font to emulate graphic objects.
- Game engine relies on a time based scheduler using a fixed time slice. No interrupts and no exceptions, except under GNU Forth 0.7.3.
- *Virtual rows* concept for scheduling on the verticals.

Presentation Layer - Part II: VT420 Font Design

Pac-Man Going Right



```
CREATE softfont
[...]
\ Character "Pacman going right", left half at $25 (%).
\ Left: columm #0 and #1 unused.
\ Group A  Group B    Group C (top to bottom)
%000000 C, %000000 C, %0000 C, \ Column #0 unused
%000000 C, %000000 C, %0000 C, \ Column #1 unused
%110000 C, %111111 C, %0000 C, \ Column #2 (eff. 0)
%110000 C, %111111 C, %0000 C, \ Column #3 (eff. 1)
%111100 C, %111111 C, %0011 C, \ Column #4 (eff. 2)
%111100 C, %111111 C, %0011 C, \ Column #5 (eff. 3)
%111111 C, %111111 C, %1111 C, \ Column #6 (eff. 4)
%111111 C, %111111 C, %1111 C, \ Column #7 (eff. 5)
%110011 C, %110000 C, %1111 C, \ Column #8 (eff. 6)
%110011 C, %110000 C, %1111 C, \ Column #8 (eff. 7)

\ Character "Pacman going right", right half at $26 (&)
\ Right: columns #8 and #9 unused.
\ Group A  Group B    Group C (top to bottom)
%001111 C, %000000 C, %1111 C, \ Column #0 (eff. 8)
%001111 C, %000000 C, %1111 C, \ Column #1 (eff. 9)
%000011 C, %000000 C, %1111 C, \ Column #2 (eff. A)
%000011 C, %000000 C, %1111 C, \ Column #3 (eff. B)
%000000 C, %000000 C, %0000 C, \ Column #4 (eff. C)
%000000 C, %000000 C, %0000 C, \ Column #4 (eff. D)
%000000 C, %000000 C, %0000 C, \ Column #4 (eff. E)
%000000 C, %000000 C, %0000 C, \ Column #4 (eff. F)
%000000 C, %000000 C, %0000 C, \ Column #8 unused
%000000 C, %000000 C, %0000 C, \ Column #9 unused

\ ---------------------------------------------------------------
\ DECDLD spec:
\ DCS Pfn; Pcn; Pe; Pcmw; Pss; Pt; Pcmh; Pcss { Dscs
\ Sxbp1 ; Sxbp2 ;...; Sxbpn ST

1  CONSTANT pfn   \ Font number
1  CONSTANT pcn   \ First soft char at $21--do not override BL!
1  CONSTANT pe    \ Erase only new definitions
10 CONSTANT pcmw  \ Character width is 10 pixels in 80 col mode
0  CONSTANT pss   \ 80 columns, 24 lines
2  CONSTANT pt    \ Full cell
16 CONSTANT pcmh  \ Character height is 16: 6/6/4 in sixels
0  CONSTANT pcss  \ 94-charset, $20 (BL) and $7E (~) are RO
85 CONSTANT ufn   \ User font name is 'U' (argument to DSCS)

\ See: https://vt100.net/docs/vt420-uu/chapter9.html
\ for full parameter description.
??oo{{~~rr/??~~~~~~oo/????BBNNNN; \ Char $25 (%) definition
NNBB??????/??????????/NNNN??????  \ Char %26 (&) definition
```

Object Oriented Approach

- Based on structures as defined in Forth 2012 (Gnu Forth 0.7.3).
- No inheritance required.
- Only two methods: **e.strategy** and **e.display**.

```
BEGIN-STRUCTURE entity
  FIELD:  e.strategy \ Strategy (moving) method
  FIELD:  e.display  \ Display method
  CFIELD: e.resurr   \ # Clock ticks till we're back (ghosts)
  CFIELD: e.reward   \ # points for killing a ghost / 100 (PM)
  CFIELD: e.vrow#    \ Virtual row number
  CFIELD: e.pcol#    \ Physical column number
  CFIELD: e.ivrow#   \ Interfering virtual row number (ghosts)
  CFIELD: e.ipcol#   \ Interfering pcol number (ghosts)
  CFIELD: e.igchr    \ Interfering grid character (ghosts)
  CFIELD: e.pcol0    \ Initial pcol number
  CFIELD: e.vrow0    \ Initial vrow number
  CFIELD: e.dir0     \ Initial direction
  CFIELD: e.hcvr#    \ Home corner vrow# (ghosts)
  CFIELD: e.hcpc#    \ Home corner pcol# (ghosts)
  CFIELD: e.cdir     \ Current direction
  CFIELD: e.pdir     \ Previous direction
  CFIELD: e.idir     \ Intended direction (PM)
  CFIELD: e.revflg   \ Reverse direction directive (ghosts)
  CFIELD: e.inited   \ TRUE if first display has been performed
  CFIELD: e.gobbling \ # Clock ticks till we're fed (PM)
  CFIELD: e.inum     \ Instance serial number
END-STRUCTURE

\ Method invokator.
: :: ( method-xt-addr -- ) @ EXECUTE ;
```

Scheduling 101

```
: _main ( -- )
  BEGIN
    remitems# @ 0= IF      \ If remitems# is 0, start new level
      .initial-grid
      update-level
      level-entry-inits
    ELSE
      \ Ghost mode handling code withdrawn.

      \ Regular entity scheduling.
      entvec #entities 0 DO
        DUP @ DUP e.strategy ::  \ Move current entity
        CELL+
      LOOP DROP

      clkperiod MS
    THEN
  AGAIN ;
```

Game's Main Engine

- A time based approach (constant time slice).
- Assumes a predictable **e.strategy** method execution time.
- Pac-Man and the ghosts are handled on an equal footing.

Business Layer Refined - Ghost Mode Management

The need for a nested scheduler emerges from the canonical specifications.

> Ghosts alternate between *scatter* and *chase* modes during gameplay at predetermined intervals. These mode changes are easy to spot as the ghosts reverse direction when they occur. Scatter modes happen four times per level before the ghosts stay in chase mode indefinitely. [. . .] The scatter/chase timer gets reset whenever a life is lost or a level is completed. At the start of a level or after losing a life, ghosts emerge from the ghost pen already in the first of the four scatter modes.

Mode	Level 1	Levels 2-4	Levels 5+
Scatter	7	7	5
Chase	20	20	20
Scatter	7	7	5
Chase	20	20	20
Scatter	5	5	5
Chase	20	1033	1037
Scatter	5	1/60	1/60
Chase	$+\infty$	$+\infty$	$+\infty$

Pinky	top left
Blinky	top right
Clyde	bottom left
Inky	bottom right

Business Layer Refined - *AI* Integration

In *chase* mode has an individual targeting strategy:

Blinky targets Pac-Man's current location.

Pinky targets a position that is four tiles ahead of Pac-Man's current moving direction.

Inky targets the end of a vector twice as long as the one originating from **Blinky** to Pac-Man's moving direction extrapolated by four half tiles.

Clyde does not know what it's doing. Its direction changes are as unpredictable as the LFSR based random number generator. The latter is, by design, completely deterministic.

This is supported by the `ghost.dirselect-nav2target` word in the game's code. In essence, for each possible next direction *dir*:

- evaluate the next coordinates assuming *dir* is taken.
- return the direction that minimizes the euclidian distance between the next coordinates and the target coordinates.

Linux and OpenVMS C Ports

POSIX based, of course, but which one?

- POSIX IEEE P1003.1-2007 (Issue 7) Draft 3: free.
- POSIX P1003.1-2024: USD 676 (USD 541 for IEEE members).

32 bit cell for compatibility with OpenVMS (possibly also OpenSolaris).

- `tcgetattr/tcsetattr` to set the terminal in raw mode.
- Basic required Forth primitives: `MS ?KEY AT-XY KEY CR`.
- OpenVMS specifics: `select` and `curses`.

Successes and Failures

VT420 configuration:

- Comms: RS-232, parity 8N1, 38400 bps unless specified otherwise.
- Terminal ID: VT220.
- Terminal mode: VT400 (7 bit controls).
- Flow control: XOFF at 64.

Situation Report:

- **Z79Forth/A 20240719** immediate real-time response.
- **GNU Forth 0.7.3/Linux native**: with `agetty` configured as VT220 and flow control specified parameters explicitely: responsive after a second or so.
- **C POSIX/Linux native** with `agetty` configured as VT220 and flow control specified parameters explicitely: responsive after a few seconds.
- **C curses/OpenVMS 9.2-3 x86_64 virtualized** flow control is not properly honored unless the line speed is dumbed down to 9600 bps. Work in progress.

Future Work and Conclusions

- Firmware supported exceptions support is needed in Z79Forth/A.
- Further work is needed on OpenVMS and OpenSolaris.
- Forth is incredibly powerful.
- Standards really do matter, especially where they are truly *free*.
- Development time can be substancially reduced by resorting to ad'hoc tools.

Q&A