```
;;
; Copyright Jacques Deschênes 2023,2024
; This file is part of stm8-gamepad
;
; stm8-gamepad is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; stm8-gamepad is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with ntsc_tuto. If not, see http://www.gnu.org/licenses/.
;;

;--------------------------------
; John H. Conway
; game of life simulation
;--------------------------------

GRID_SIDE=20
GRID_SIZE=(GRID_SIDE*GRID_SIDE)/8
GRID_LEFT=(CHAR_PER_LINE-GRID_SIZE)/2
GRID_TOP=2
GRID_CNTR_X=GRID_LEFT+(GRID_SIDE)/2
GRID_CNTR_Y=GRID_TOP+(GRID_SIDE)/2
CELL=133
EMPTY=SPACE
```

PROF

```
.area G_DATA (ABS)
.org 4
```

```
gy: .blkb 1; y coord
gx: .blkb 1 ; x coord
gen: .blkw 1 ; generation
src: .blkw 1 ; source grid address
dest: .blkw 1 ; dest grid address
grid1: .blkb GRID_SIZE
grid2: .blkb GRID_SIZE
```

```
.area CODE
```

```
print_gen:
pushw x
_ldxz gen
ldw x,#12
_strxz cy
_ldxz gen
call put_uint16
popw x
ret

life_init:
call tv_cls
ldw y,#gen_str
call tv_puts
clr a
_straz gen
_straz gen+1
ldw x,#2*GRID_SIZE
ldw y,#grid1
_stryz src
call fill
ldw x,#grid2
_strxz dest
_movz gx,GRID_CNTR_X
_movz gy,GRID_CNTR_Y
ret

gen_str: .asciz "GENERATION: "

;------------------
; set cell
; at gx,gy position
; input:
; A character
;------------------
write_cell:
_ldxz gy
addw x,#(GRID_LEFT<<8)+GRID_TOP
_strxz cy
call tv_putc
ret

;-------------------------
; toggle cell state
; input:
```

```
; XL gx
; XH gy
;--------------------------
toggle_cell:
```

```
  ret
```

```
;--------------------------
; compute next generation
;--------------------------
next_gen:
```

```
  ret
```

```
;---------------------
; user initialize grid
; moving cursor around
;---------------------
pattern_init:
ld a,#128
call write_cell
call wait_key
push a
ld a,#SPACE
call write_cell
ld a,#BTN_UP
cp a,(1,sp)
jrne 2$
tnz gy
jreq pattern_init
_decz gy
jra pattern_init
2$: ld a,#BTN_DOWN
cp a,(1,sp)
jrne 4$
ld a,#GRID_SIDE-1
cp a,gy
jreq pattern_init
_incz gy
jra pattern_init
4$: ld a,#BTN_LEFT
cp a,(1,sp)
jrne 6$
```

```
tnz gx
jreq pattern_init
_decz gx
jra pattern_init

6$: ld a,#BTN_RIGHT
cp a,(1,sp)
jrne 8$

8$: ld a,#BTN_A
cp a,(1,sp)
jrne 12$
call toggle_cell
jra pattern_init
12$: ld a,#BTN_B
cp a,(1,sp)
jrne pattern_init
call wait_key_release
_drop 1
ret

game_of_life:
call life_init
call pattern_init
2$: call print_gen
call next_gen
call kpad_input
jreq 2$
call tv_cls
ret
```