|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | P1 | P1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | P1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | BL |  |  |  | BL |  |  |  |  | BL |  |  |  |  |
|  |  |  |  | OCK |  |  |  |  | OCK |  |  |  | OCK |  |  |  |
|  |  |  |  |  | BO |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  | MB |  |  |  |  |  |  |  |  |  |  |
|  |  |  | BL |  |  |  | BL |  |  |  |  | BL |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | OCK |  |  |  |  | OCK |  |  |  | OCK |  |  |  |
|  |  | BO |  |  | EX | PL |  |  |  |  |  |  |  |  |  |  |
|  | MB |  |  |  | OS | ION |  |  |  |  |  |  |  |  |  |  |
|  |  |  | BL |  | EX | PL | BL |  |  |  |  | BL |  |  |  |  |
|  |  |  |  | OCK | OS | ION |  |  | OCK |  |  |  | OCK |  |  |  |
|  | EX | PL | EX | PL | EX | PL | EX |  | PL | EX | PL |  |  |  | P2 |  |
|  | OS | ION | OS | ION | OS | ION | OS |  | ION | OS | ION |  |  | P2 | P2 |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Example monome grid layout. All coloured grids are lit LEDs (regardless of colour).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15, 15 | 15, 14 | 15, 13 | 15, 12 | 15, 11 | 15, 10 | 15, 9 | 15, 8 |  | 15, 7 | 15, 6 | 15, 5 | 15, 4 | 15, 3 | 15, 2 | 15, 1 | 15,0 |
| 14, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 13, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 12, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 11, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 10, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 0, 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0, 0 |

Splash screen for BomberNome, spells out “BOMB NOME”.

DEBUG\_3 – PD.1

DEBUG\_2 – PD.0

DEBUG\_1 – PA.4

DEBUG\_0 – PA.3

Firefly 2.2

3-8 DECODER

3-8 DECODER

3-8 DECODER

3-8 DECODER

3-8 DECODER

3-8 DECODER

3-8 DECODER

3-8 DECODER

INVERTER

INVERTER

INVERTER

INVERTER

Monome #1

Monome #2

Monome #3

Monome #4

P1

P1

PORTB[7:0]

PTAD[7:0]

PTT[7:0]

PORTA[7:0]

HCS12: MC9S12C128

PTM[5:0]

BUZZER

bmac

BomberNome Connections and Hardware Diagram

Read X, Y, NRK\_BUTTON

ACTION = BOMB

Y > MAX\_ THRESH

Y < MIN\_ THRESH

X > MAX\_ THRESH

X < MIN\_ THRESH

NRK\_ BUTTON

ACTION =  
UP

ACTION = DOWN

ACTION = RIGHT

ACTION =  
LEFT

ACTION =  
NONE

YES

Wait for DONE signal

YES

YES

YES

YES

NO

NO

NO

NO

NO

Player FireFly  
Flow Diagram

Broadcast ACTION, ID\*

\*ID is hardcoded for each FireFly, ID=0 for Player 1, ID=1 for Player 2

Receive ACTION, ID

DEBUG = 0b0101

UP

DOWN

RIGHT

LEFT

BOMB

DEBUG = 0b0001

DEBUG = 0b0010

DEBUG = 0b0100

DEBUG = 0b0011

DEBUG = 0b0000

YES

YES

YES

YES

YES

NO

NO

NO

NO

NO

Master FireFly  
Flow Diagram

Update DEBUG output pins

ID = 1

DEBUG |= 0b1000

NO

YES

Wait for next package

Read PORTA, PORTB

NO

BomberNome Input  
Flow Diagram

input =  
((~PORTA & 0xC0) >> 6) | ((~PORTB & 0xC0) >> 4)

Player dying

Player dead

Update player actions for next game state update

Turn on buzzer:  
PTM = 0xFF

Turn off buzzer:  
PTM = 0x00

update\_actions()

return

NO

YES

YES

BomberNome Display  
Flow Diagram

Update led\_grid[16][16] based on game\_grid[7][7]

Update lit led lists for each monome

update\_display()

Infinite loop

Select next led in lit led list

Set PORTA, PORTB, PTAD, PTT

Delay for 100 CPU cycles

return

Interrupt

Main

Interrupt

BomberNome Gameplay  
Flow Diagram

update\_game()

Create new game\_grid[7][7]

return

Check player actions

Update player bombs

Draw bombs onto game\_grid

Draw players onto game\_grid

Bomb and < 3 bombs out

Move and no wall

Move player’s location

No change (draw in same location)

Place new bomb at location

NO

NO

YES

YES

De-active bomb

No bomb animation

NO

Update bomb counter

Bomb exploding

Check for players in range

Player in range

Set player action to dying

YES

YES

Bomb flashing animation

NO

YES

Bomb exploding animation

No change (keep player actions)

NO

BomberNome Overall  
Flow Diagram

P1 & P2 BOMB

Start

Splash Screen

Setup and start timer

Update screen based on game state (infinite loop)

Timer interrupt

0.33s interval

update\_game() update\_display()

Read input

Game running

Player died

NO

NO

NO

NO

YES

YES

YES

YES

update\_actions()