uOttawa

L'Université canadienne
Canada's university

CEG3136
Computer Architecture II

Introduction to Parallel Ports

Notes from

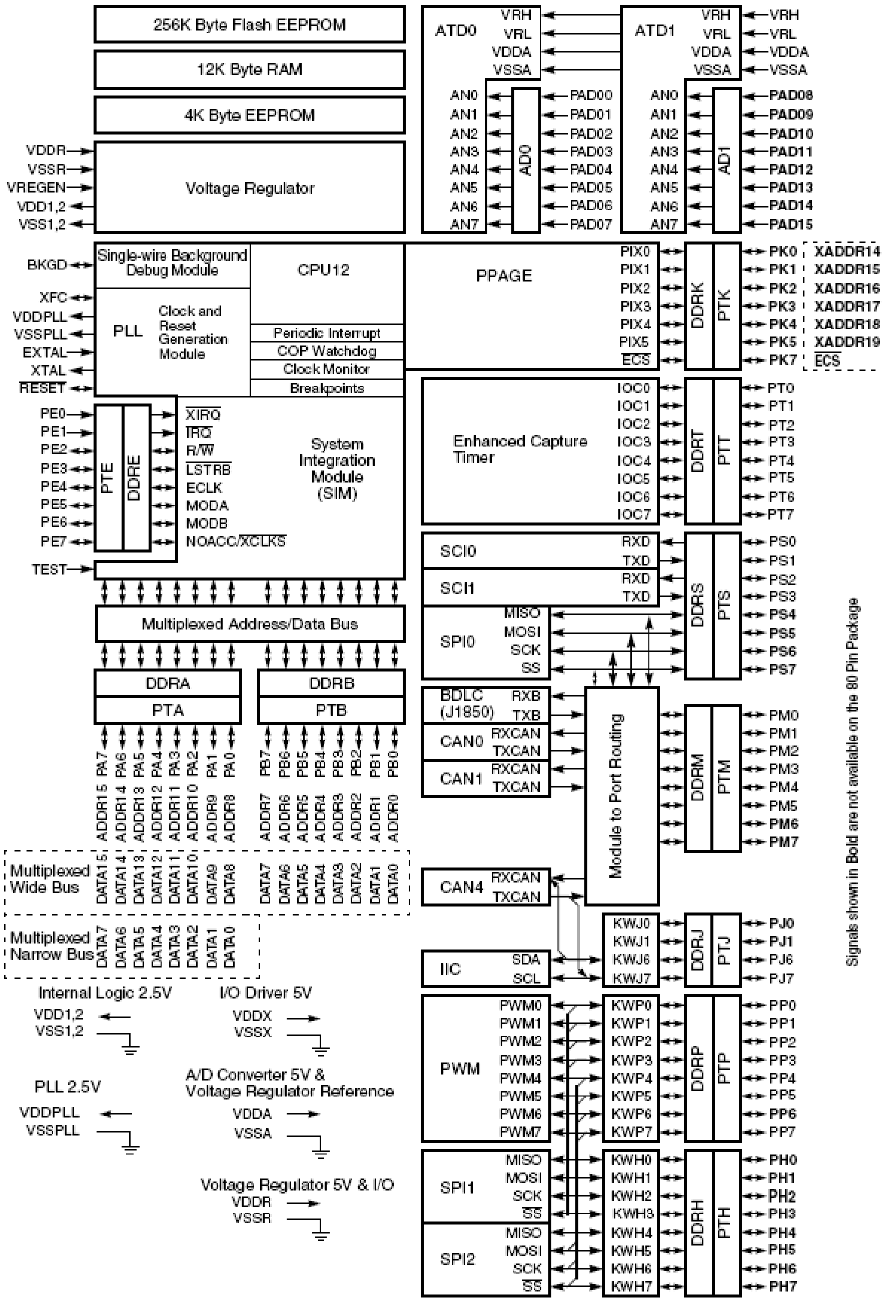
Dr. Voicu Groza

Université d'Ottawa | University of Ottawa

www.uOttawa.ca

MC9S12DG256 Block Diagram

Figure 1-1 MC9S12DT256 Block Diagram



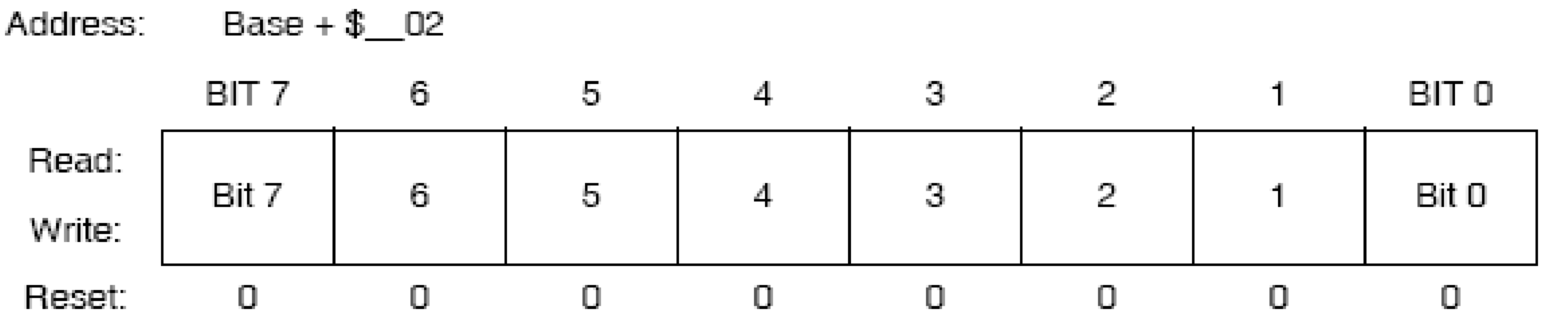
Overview of the 9S12DG256 Parallel Ports

- Different members of the HCS12 family contains from 80 to 112 pins
- Pins can be used for different functions
- The MC9S12DG256 offers a number of parallel ports in single chip mode: A, B, E, and K
  - □ Other ports can also be used for parallel I/O (e.g. Port P)

Ports A, B, E and K

- Parallel port registers
  - Data Direction Registers: DDRA, DDRB, DDRE, DDRK
    - Defines pins as input or output
  - Data Registers: PORTA, PORTB, PORTE, PORTK
    - Used to determine values of input pins or set the levels of the output pins.
  - PUCR Register
    - On bit per port to activate the pull up registers

Data Direction Register: Port A



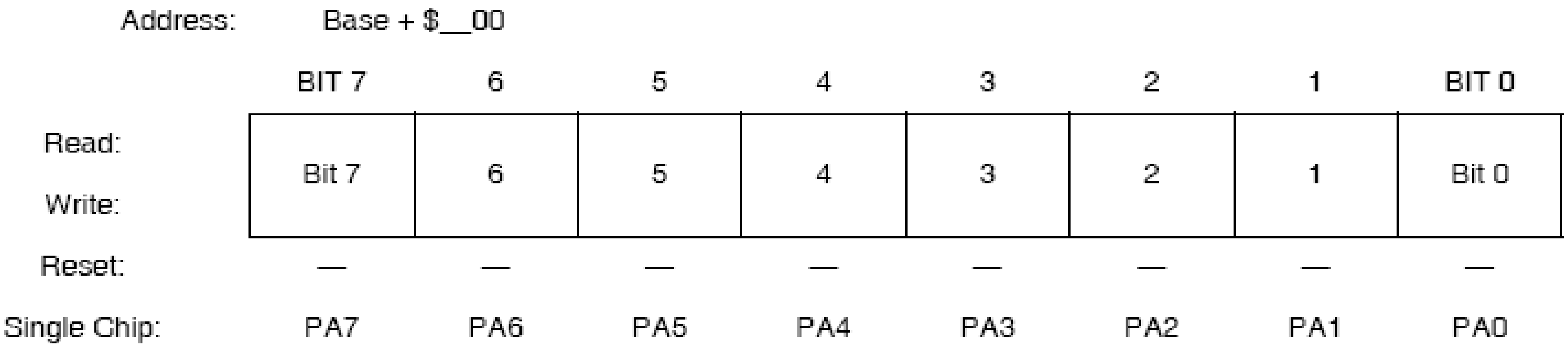
Address: Base + $__02

	BIT 7	6	5	4	3	2	1	BIT 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 3-4 Data Direction Register A (DDRA)

- Each bit corresponds to a pin of the port
- Configure the port pins
  - □ 0 ⇒ the pin corresponding to the bit is configures as an input pin
  - □ 1 ⇒ the pin corresponding to the bit is configured as an output pin

Data Register: Port A

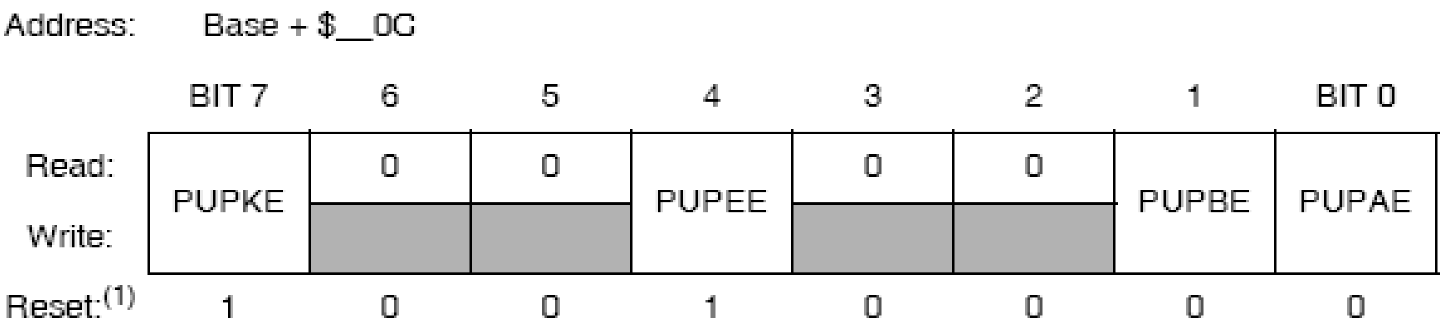


Address: Base + $__00

	BIT 7	6	5	4	3	2	1	BIT 0
Read: Write:	Bit 7	6	5	4	3	2	1	Bit 0
Reset:	—	—	—	—	—	—	—	—
Single Chip:	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0

- Each bit corresponds to a pin of the port
- Reading:
  - Gives the state of the pin, 0V ⇒ 0, 5V ⇒ 1
- Writing:
  - For output pins, sets the level on the pins:
    0 ⇒ 0V, 1 ⇒ 5V
  - No effect on input pins

The PUCR



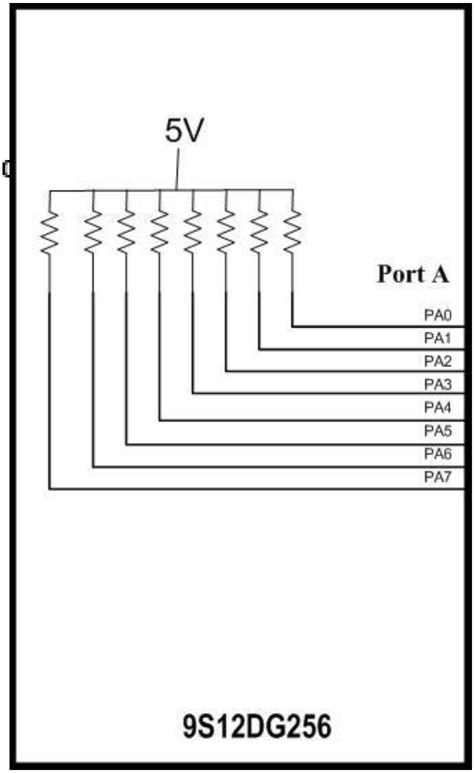
Address: Base + $__0C

	BIT 7	6	5	4	3	2	1	BIT 0
Read:	PUPKE	0	0	PUPEE	0	0	PUPBE	PUPAE
Write:		▨	▨		▨	▨		
Reset:[1]	1	0	0	1	0	0	0	0

▨ = Unimplemented

NOTES:
1. The default value of this parameter is shown. Please refer to the specific devi
Guide to determine the actual reset state of this register.

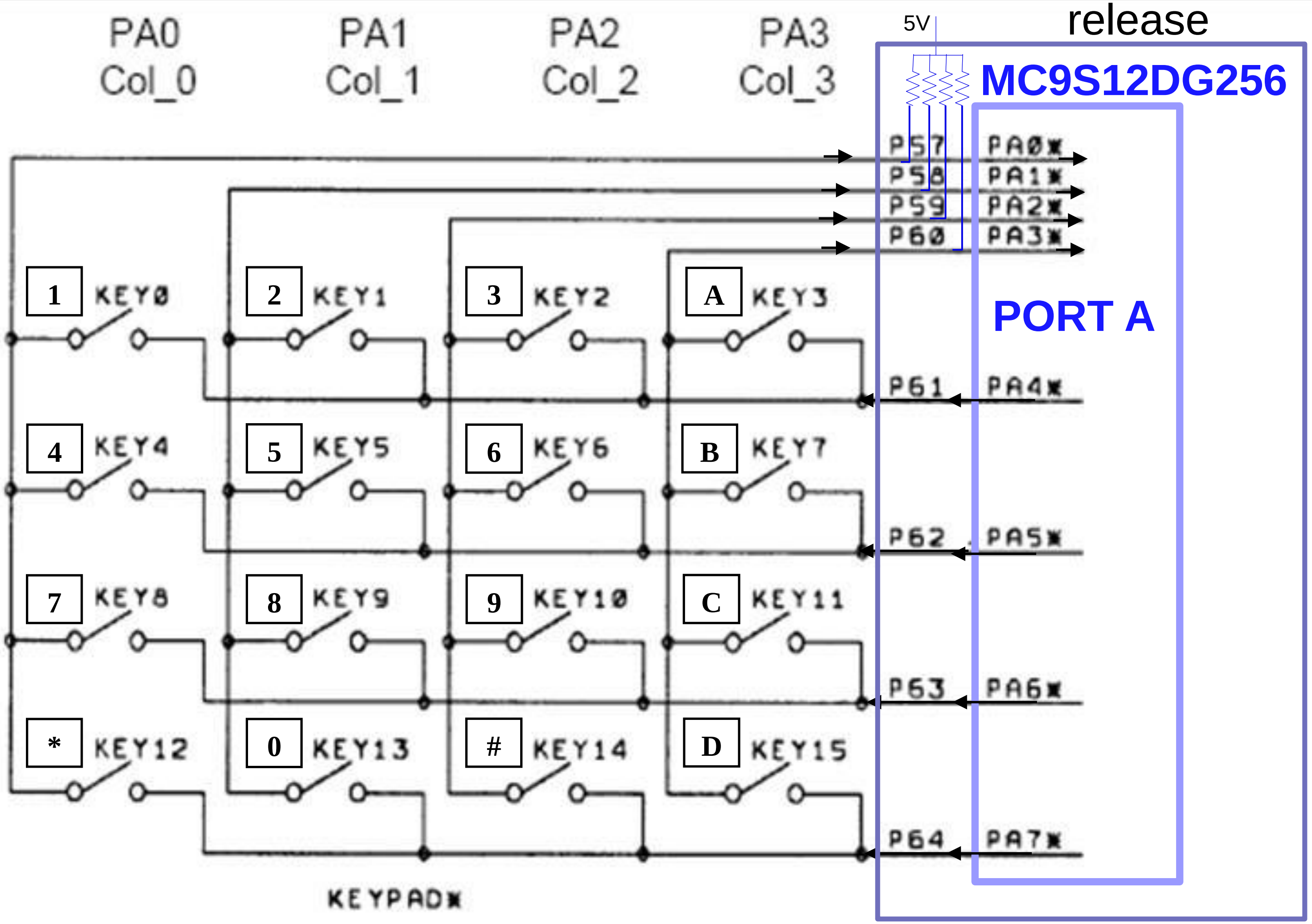
Figure 3-11 Pullup Control Register (PUCR)



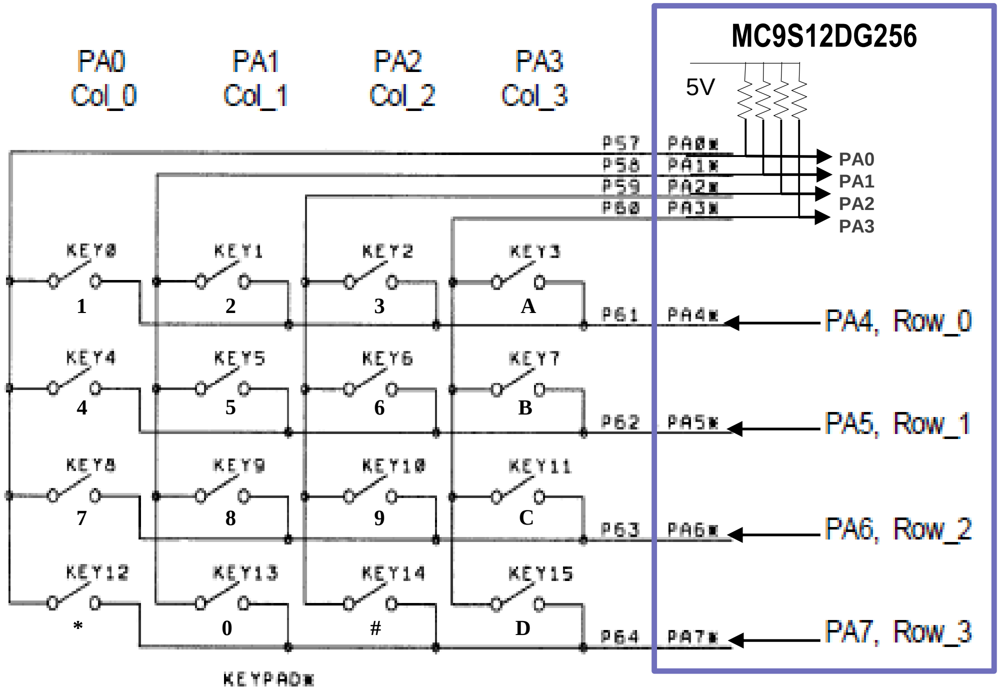
- When the PUCR bit is set, pullup resistors will be activated for the corresponding port (for input)
- No effect on output pins

Dragon-12 Key Pad

1. Wait for a keypress
2. Detect a keypress and debounce
3. Extract the keypress code
4. Translate to an ASCII value
5. Detect and debounce the key release



DRAGON12 Plus Trainer



Keypad scan routine **sets PA7 low & PA4, PA5, PA6 high (PA= 0111 XXXX), then tests PA0-PA3:**

- If no key is down, PA0-PA3 remain high (0111 **1111**)
- If PA3 = low (PA= **0111 0111**), the key 15 is down.
- If PA2 = low (PA= **0111 1011**), the key 14 is down.
- If PA1 = low (PA= **0111 1101**), the key 13 is down.
- If PA0 = low (PA= **0111 1110**), the key 12 is down.

Keypad scan routine **sets PA6 low and PA4, PA5, PA7 high, (PA= 1011 XXXX), then tests PA0-PA3:**

- If no key is down, PA0-PA3 remain high.
- If PA3 = low (1011 0111) the key 11 is down
- If PA2 = low (1011 1011) the key 10 is down
- If PA1 = low (1011 1101) the key 9 is down
- If PA0 = low (1011 1110) the key 8 is down.

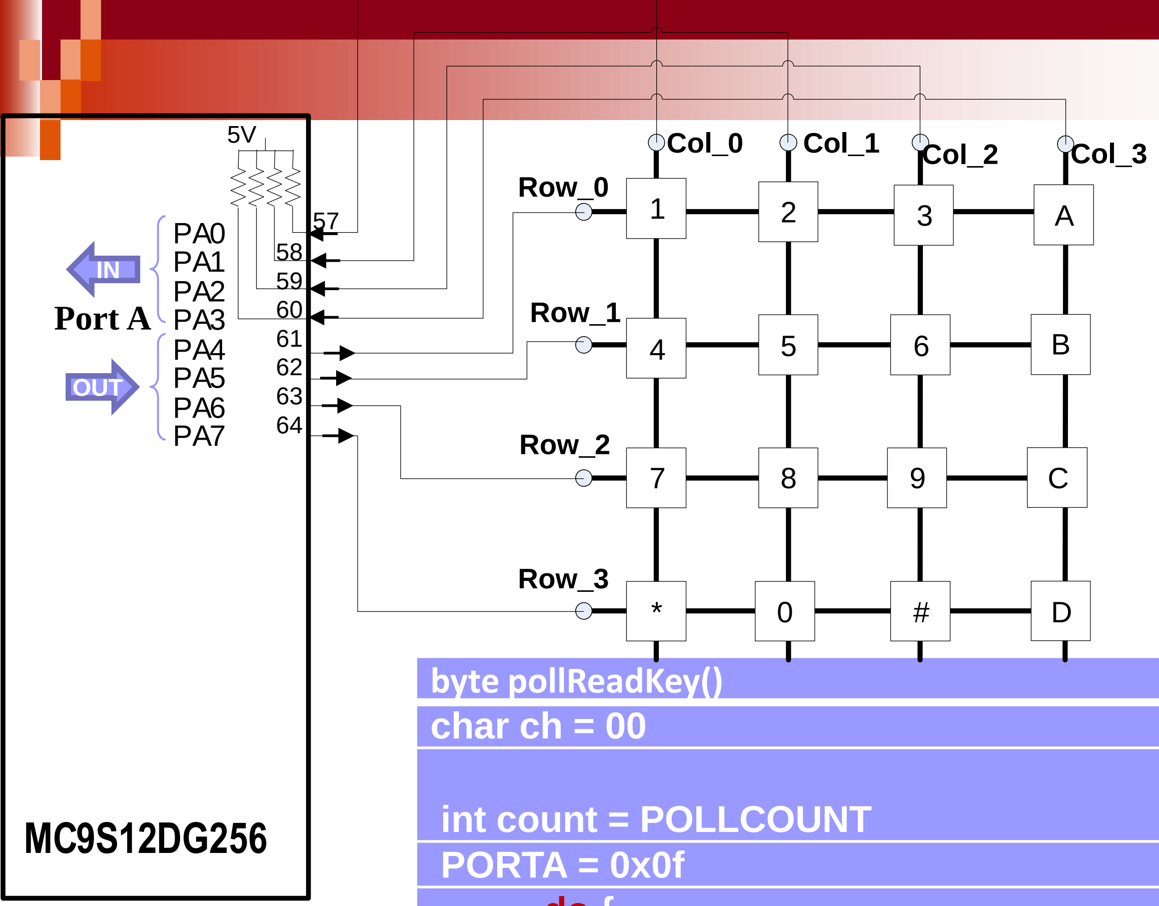
Keypad scan routine **sets PA5 low and PA4, PA6, PA7 high, then tests PA0-PA3:**

- If no key is down, PA0-PA3 remain high.
- If PA3 = low, the key 7 is down.
- If PA2 = low, the key 6 is down.
- If PA1 = low, the key 5 is down.
- If PA0 = low, the key 4 is down.

Keypad scan routine **sets PA4 low and PA5, PA6, PA7 high, then tests PA0-PA3:**

- If no key is down, PA0-PA3 remain high.
- If PA3 = low, the key 3 is down.
- If PA2 = low, the key 2 is down.
- If PA1 = low, the key 1 is down.
- If PA0 = low, the key 0 is down.

uOttawa



pollReadKey

checks if a key has been pressed and returns its value; otherwise returns a null character (0).

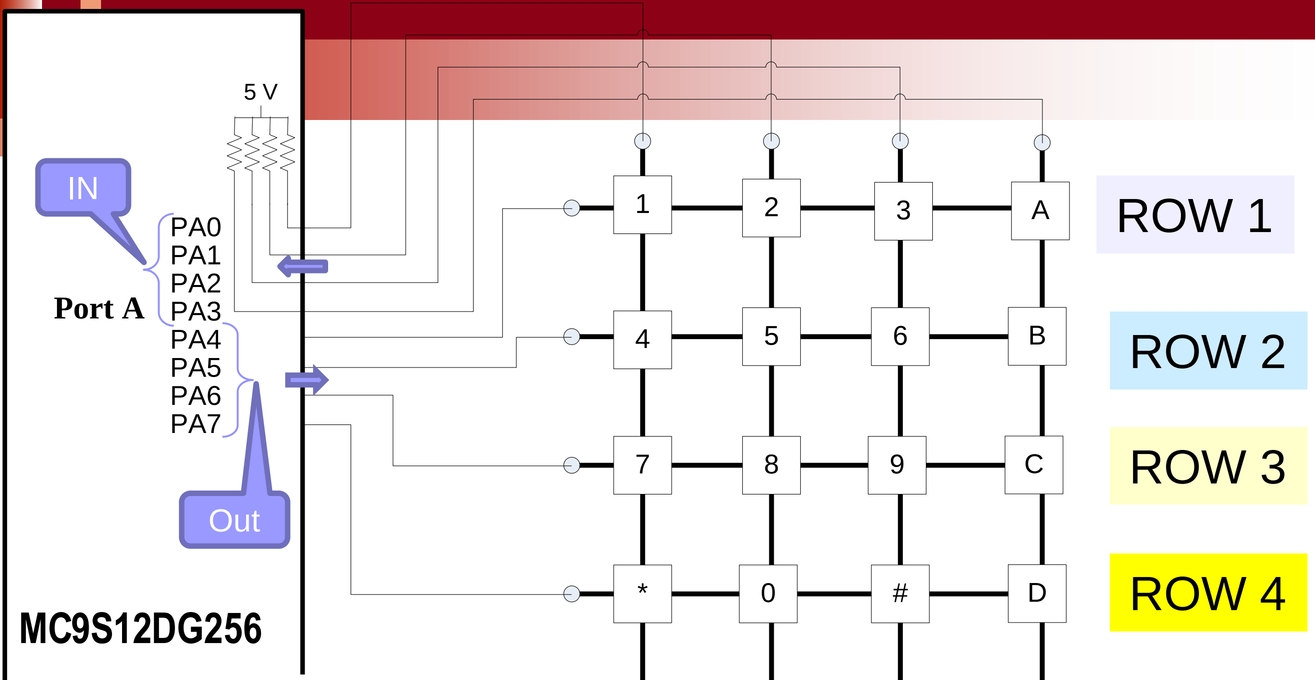
```
byte pollReadKey()
char ch = 00                          // NOKEY = 00

 int count = POLLCOUNT                // POLLCOUNT = 1ms
 PORTA = 0x0f                         //set PA7-PA4  outputs to low
          do {
                if(PORTA != 0x0f)     // any key pressed?
                 {
                      delayms(1)
                      if(PORTA != 0x0f)   // was the key really pressed?
                      {
                          ch = readKey();   // get the code of the pressed key
                          break;
                       }
                     }
               count--;
             } while(count != 0);
```

readKey

What does it do:

- Reads a key from the keypad (returns the ASCII code of the key pressed).
- Debouncing both pressing and releasing of the key
- The value of the key is returned once the key is released

byte code	
byte ch	
do	
{	
PORTA = 0x0F	// set all output pins PA_7-PA_4 to 0
while(PORTA==0x0F);	// check for the leading edge (all input bits PA_3 -PA_0 are 1 until a key is pressed)
code = PORTA;	// get the keycode
delayms(10);	// Debouncing pressing of the key
} while(code != PORTA);	// start again if PORTA has changed
code = readKeyCode();	// call subroutine **readKeyCode** to get the keycode
PORTA = 0x0F;	// set all output pins PA_7-PA_4 to 0
while(PORTA!=0F) ;	// wait for the trailing edge
delayms(10);	// Debouncing release of the key
ch = translate(code);	// call subroutine to **translate** code to ASCII
return(ch);	



Read Key Code

Detects the pressed key and returns its **key code**.

```
ROW1 EQU %1110 1111
ROW2 EQU %1101 1111
ROW3 EQU %1011 1111
ROW4 EQU %0111 1111
PORTA = ROW1
 if(PORTA == ROW1) //key pressed is not in ROW1
 {
   PORTA = ROW2        // maybe in ROW2?
   if(PORTA == ROW2) //key pressed is not in ROW2
    {
      PORTA = ROW3       // maybe in ROW3?
      if(PORTA == ROW3) //key pressed is not in ROW3
         PORTA = ROW4 //then it has to be in ROW4
    }
 }
 key = PORTA
 return(key)
```

Key Code		Key
OUT	IN	ASCII
PA_7-PA_4	PA_3-PA_0	
XXXX	1111	none
1110	1110	'1'
1110	1101	'2'
1110	1011	'3'
1110	0111	‘A'
1101	1110	'4'
1101	1101	'5'
1101	1011	'6'
1101	0111	‘B'
1011	1110	'7'
1011	1101	'8'
1011	1011	'9'
1011	0111	‘C'
0111	1110	'*'
0111	1101	'0'
0111	1011	'#'
0111	0111	‘D'