SISTEMI EMBEDDED

Programming the DE2 Basic Computer:
playing with parallel ports

Federico Baronti

Last version: 20170314

Putting into practice (1a)

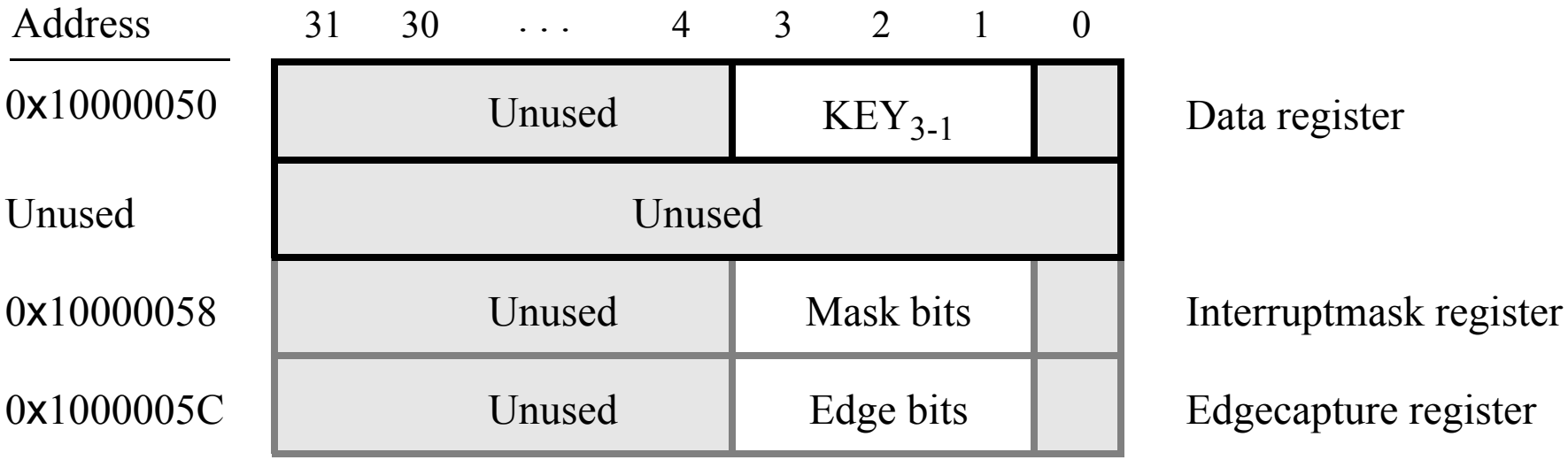
- Write a program that turns on a single green led among LEDG7-LEDG0 and makes the position of the on-LED rotate with a period of around 500 ms. Make the activation and direction of the rotation controllable by the pushbuttons KEY3-KEY1, as follows:
  - KEY2 stops rotation, KEY3 and KEY1 activate rotation clockwise and counterclockwise respectively
  - The program must be sensitive to the edges originated by the pressure of the pushbuttons KEY3-KEY1

Putting into practice (1b)

- Hints:
  - Recognize pushbutton activations through the EVENT register of the relevant Parallel Port
  - Store the LEDG7-LEDG0 status on a 8-bit unsigned variable
  - Use <<, >> for left and right rotation (be careful to manage the all-zero situation)
  - Use a finite state machine (Moore model) to:
    - update the rotation state according to the KEY3-KEY1 events
    - generate the new LEDG7-LEDG0 status through a *switch* instruction that scans the rotation state

Putting into practice (1c)

- Hints: Pushbutton Parallel Port



- To clear the *Edgecapture register* write any value into it (it is cleared all at once)
  - This behavior is specific of the Parallel Port peripheral and may be different in other peripherals

Putting into practice (1d)

- Hints:
  - Use the *Wait_ms()* function to generate the rotation period

```
/* delay generation */
#define CYCLES_PER_MS 254
/* value hand tuned to achieve around 1 ms resolution
*  for DE2 Basic Computer (code optimization OFF)
*/
void Wait_ms(unsigned int time_ms) {
int i,j;
        for(j=0; j<time_ms; j++) {
                for(i=0; i<CYCLES_PER_MS; i++) {;}
        }
}
```

  - What does it happen if the LEDG7-LEDG0 status is stored in a signed variable?

Putting into practice (2)

- **Faster click game**:
  - Detect which of KEY1 and KEY3 is pressed first after the turning on one of the GREEN LEDS
  - Make the interval time between two consecutive switching on of the LED random
  - Make also the GREEN LED position random
  - Signal which KEY has been pressed first using two different RED LEDs
  - Display the number of times KEY3 has been pressed first on HEX3-HEX2 and KEY1 on HEX1-HEX0
  - Use one SLIDER to start/stop the game and reset the scoring

Putting into practice (3a)

- **Week day**
  - Show on GREEN LEDS 6..0 the day of the week of an arbitrary date after 1582 (Gregorian calendar), which is set using KEY3..1 and displayed on the 7-seg displays
  - Use KEY3 to move circularly from day to month to year and KEY2 and KEY1 to change the selected digit of the date
  - Use the RED LEDs below the corresponding 7-seg to indicate the selected digit of the date

Putting into practice (3b)

- **Week day**
  - How a C++ program for a PC using standard I/O streams looks looks like:

```
// gionosett.cpp
#include <iostream>
using namespace std;
int main(){
   int giorno, mese, anno, sett;
   cout << " Scrivi una data nel formato giorno mese anno\n";
   cin >> giorno >> mese >> anno;
   if (mese <= 2) {
      sett = (anno+31*(mese-1)+giorno+(anno-1)/4-
3*((anno+99)/100)/4)%7; }
   else {
      sett = (anno+31*(mese-1)+giorno-(4*mese+23)/10+anno/4-
(3*(anno/100+1)/4))%7;  }
// …
```

Putting into practice (3c)

- **Week day**
  - How a C++ program for a PC using standard I/O streams looks like:

```
cout << "Il giorno " << giorno << '/' << mese << '/' << anno << " cade di ";
   switch(sett) {
      case 0: cout << "sabato\n"; break;
      case 1: cout << "domenica\n"; break;
      case 2: cout << "lunedi`\n"; break;
      case 3: cout << "martedi`\n"; break;
      case 4: cout << "mercoledi`\n"; break;
      case 5: cout << "giovedi`\n"; break;
      case 6: cout << "venerdi`\n"; break;
      }
```