**SAKLAR dan LED**

1. **Tujuan**

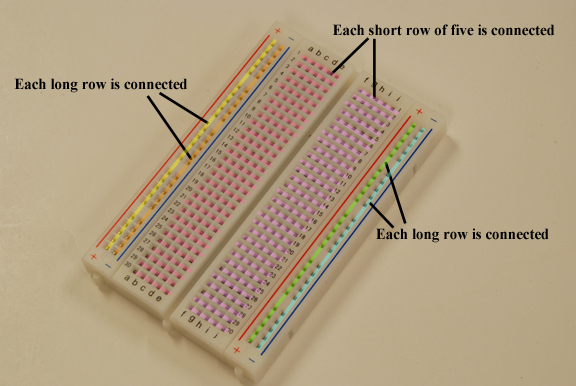
* Memahami unsur-unsur rangkaian dasar seperti source, ground, dan resistor.
* Memahami bagaimana saklar dan LED bekerja.
* Aplikasi Hukum Ohm.
* Desain rangkaian analog dan konstruksi di protoboard.
* Antarmuka saklar dan LED ke mikrokontroler.
* Pemrograman logika sederhana.

1. **Peralatan yang digunakan**

* Keil μVision v5
* Tiva C Series LaunchPad
* Stellaris® ICDI Drivers
* Baterai
* Resistor 470 Ω
* Saklar
* LED
* Protoboard
* Kabel male-female
* Resistor 10 kΩ

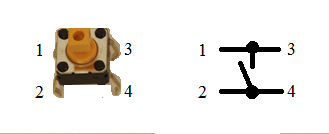
1. **Landasan teori**

Untuk membangun rangkaian, kita akan menggunakan protoboard. Lubang-lubang dalam protoboard terhubung internal secara sistematis, seperti terlihat dalam Gambar 1.



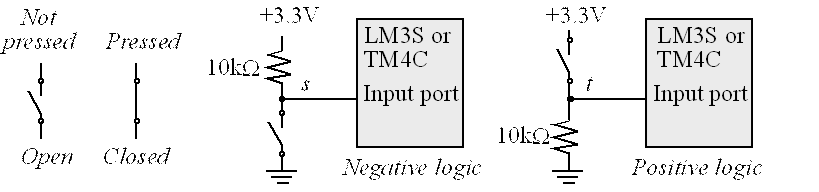
Gambar 1. Struktur protoboard

Baris panjang lubang-lubang sepanjang sisi terluar protoboard terhubung secara elektris. Beberapa protoboard seperti dalam Gambar 1 memiliki empat baris panjang (dua di setiap sisi), sementara yang lain hanya memiliki dua baris panjang (satu di setiap sisi). Kita menggunakan baris panjang sebagai bus daya. Jika protoboard hanya memiliki dua baris panjang (satu di setiap sisi), kita akan menghubungkan satu baris ke +3,3 V dan baris lain ke ground. Jika protoboard memiliki dua baris panjang di setiap sisi, maka dua baris ke ground, satu baris ke +3,3 V. Gunakan spidol hitam dan label tegangan pada setiap baris. Di tengah protoboard, akan ditemukan dua kelompok lubang 0,1 inci. Setiap baris yang berdekatan dari lima lubang terhubung secara elektris. Kita biasanya menyisipkan komponen ke dalam lubang. Jika rangkaian terpadu (IC) akan ditempatkan ke protoboard, hal itu dilakukan sedemikian rupa sehingga dua baris pin di tengah celah.



Gambar. Saklar B3F

Perangkat input/output adalah komponen kritis sistem embedded. Perangkat input pertama yang kita pelajari adalah saklar. Hal ini memungkinkan manusia memasukkan informasi biner ke dalam komputer. Biasanya kita mendefinisikan keadaan tegas, atau logika benar, ketika saklar ditekan. Saklar kontak dapat juga digunakan dalam mesin untuk mendeteksi kontak mekanis (misal dua bagian menyentuh, kertas ada dalam printer, atau roda di tanah). Saklar single pole single throw (SPST) memiliki dua koneksi. Saklar diperlihatkan sebagai lingkaran terbuka kecil dalam Gambar 2.



Gambar 2. Antarmuka saklar Single Pole Single Throw (SPST)

Pada saklar normally open (NO), resistansi antar sambungan tak terhingga (melebihi 100 MΩ pada saklar B3F) jika saklar tidak ditekan. Resistansi menjadi nol (dibawah 0,1 Ω pada saklar B3F) jika saklar ditekan. Untuk mengubah resistansi takhingga/nol menjadi sinyal digital, kita dapat menggunakan resistor pull-down ke ground atau resistor pull-up ke +3,3 V seperti terlihat dalam Gambar 2. Perhatikan bahwa 10 kΩ adalah 100.000 kali lebih besar daripada resistansi-on saklar dan 10.000 kali lebih kecil daripada resistansi-off saklar. Cara lain untuk memilih resistor pull-down atau pull-up adalah mempertimbangkan arus input ke pin input mikrokontroler. Arus ke dalam mikrokontroler akan kurang dari 2 μA (diperlihatkan sebagai IIL dan IIH dalam datasheet). Jadi, jika arus ke dalam mikrokontroler sebesar 2 μA, maka penurunan tegangan sepanjang resistor 10 kΩ akan menjadi 0,02 V, yaitu dapat diabaikan. Dengan resistor pull-down, sinyal digital akan rendah jika saklar tidak ditekan dan tinggi jika saklar ditekan (kanan Gambar 2). Ini didefinisikan sebagai **logika positif** karena keadaan tegas adalah logika tinggi. Sebaliknya, dengan resistor pull-up, sinyal digital akan tinggi jika saklar tidak ditekan dan rendah jika saklar ditekan (tengah Gambar 2). Ini didefinisikan sebagai **logika negatif** karena keadaan tegas adalah logika rendah.

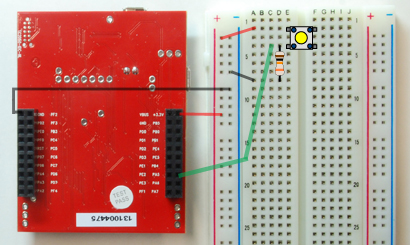
Salah satu masalah rumit dengan saklar mekanis adalah osilasi on dan off ketika ditekan dan dilepas. Osilasi kontak bervariasi dari saklar 1 ke saklar lain dan dari waktu 1 ke waktu lain, tapi biasanya osilasi adalah kejadian sementara yang berlangsung kurang dari 5 ms. Kita dapat menghilangkan efek osilasi jika kita mendesain perangkat lunak yang menunggu paling sedikit 10 ms antara waktu kita membaca nilai saklar.

Untuk antarmuka saklar, kita menghubungkannya ke pin dan inisialisasi pin sebagai input (Gambar 3).



Gambar 3. Antarmuka saklar ke input mikrokomputer

Fungsi inisialisasi akan enable clock, mengatur register direction ke input, mematikan fungsi alternative, dan enable pin. Perhatikan perangkat lunak friendly karena hanya mempengaruhi PA5 tanpa mempengaruhi bit lain dalam Port A. Fungsi input membaca Port A dan mengembalikan nilai benar (0x20) jika saklar ditekan dan mengembalikan nilai salah (0x00) jika saklar tidak ditekan. Gambar 4 memperlihatkan bagaimana kita dapat membangun rangkaian ini dengan protoboard dan LaunchPad.



Gambar 4. Konstruksi antarmuka saklar ke input mikrokontroler

Perangkat lunak dalam Program 1 disebut **driver**, dan mencakup inisialisasi yang dipanggil sekali, dan fungsi kedua yang dapat dipanggil untuk membaca posisi saklar.

|  |
| --- |
| **#define PA5   (\*((volatile unsigned long \*)0x40004080)) void Switch\_Init(void){ volatile unsigned long delay;   SYSCTL\_RCGC2\_R |= 0x00000001;     // 1) activate clock for Port A   delay = SYSCTL\_RCGC2\_R;           // allow time for clock to start                                     // 2) no need to unlock GPIO Port A   GPIO\_PORTA\_AMSEL\_R &= ~0x20;      // 3) disable analog on PA5   GPIO\_PORTA\_PCTL\_R &= ~0x00F00000; // 4) PCTL GPIO on PA5   GPIO\_PORTA\_DIR\_R &= ~0x20;        // 5) direction PA5 input   GPIO\_PORTA\_AFSEL\_R &= ~0x20;      // 6) PA5 regular port function   GPIO\_PORTA\_DEN\_R |= 0x20;         // 7) enable PA5 digital port } unsigned long Switch\_Input(void){   return PA5; // return 0x20(pressed) or 0(not pressed) } unsigned long Switch\_Input2(void){   return (GPIO\_PORTA\_DATA\_R&0x20); // 0x20(pressed) or 0(not pressed) }** |

Program 1. Antarmuka perangkat lunak untuk saklar pada PA5 (C8\_Switch)

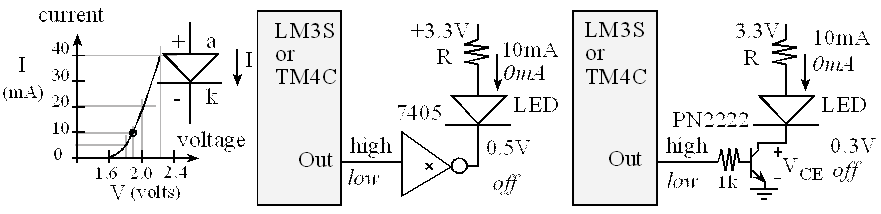
Penulisan perangkat lunak seperti ini disebut **abstraction, karena memisahkan apa yang saklar lakukan (**Init, On, Off) dari cara kerjanya (PortA, bit 5, TM4C123). Fungsi input pertama menggunakan alamat bit-khusus hanya untuk mendapatkan PA5, sedangkan fungsi kedua membaca seluruh port dan memilih bit 5 menggunakan logika AND.

**Light emitting diode** (LED) memancarkan cahaya ketika arus listrik melewatinya. LED memiliki polaritas, bermakna arus harus lewat dari anoda ke katoda untuk aktivasi. Anoda diberi label **a** atau **+**, dan katoda diberi label **k** atau **-**. Katoda adalah kaki pendek dan anoda adalah kaki lebih panjang.



Gambar. LED

Kecerahan LED bergantung pada daya listrik (P = I × V). Berhubung tegangan LED kira-kira konstan dalam daerah aktif (lihat kiri Gambar 5), kita dapat membuat kecerahan yang diinginkan dengan mengatur arus.

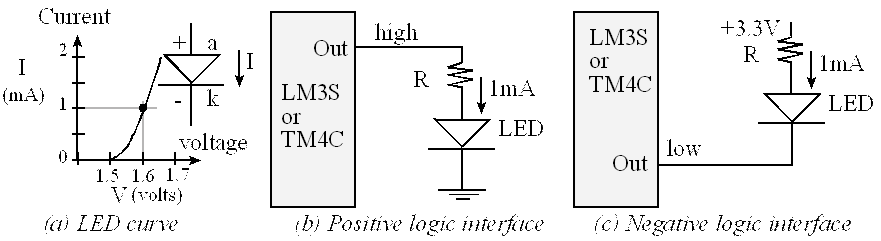


Gambar 5. Antarmuka LED logika positif

Jika arus LED diatas 8 mA, kita tidak dapat menghubungkannya langsung ke mikrokontroler karena arus tinggi dapat merusak chip. Gambar 5 memperlihatkan dua rangkaian antarmuka yang mungkin dapat digunakan. Dalam kedua rangkaian, jika perangkat lunak membuat output-nya tinggi, LED akan hidup. Jika perangkat lunak membuat output-nya rendah, LED akan mati (ditunjukkan pada Gambar 5 dengan *italics*). Ketika perangkat lunak menulis logika 1 ke port output, input ke 7405/PN2222 menjadi tinggi, output dari 7405/PN2222 menjadi rendah, 10 mA berjalan melalui LED, dan LED hidup. Ketika perangkat lunak menulis logika 0 ke port output, input ke 7405/PN2222 menjadi rendah, output dari 7405/PN2222 float (tidak tinggi atau rendah), tiada arus berjalan melalui LED, dan LED mati. Nilai resistor dipilih untuk membuat arus LED yang tepat. Ketika aktif, tegangan LED menjadi sekitar 2 V, dan daya dikirim ke LED akan dikendalikan oleh arusnya. Jika kecerahan yang diinginkan membutuhkan titik operasi 1,9 V pada 10 mA, maka nilai resistor sebaiknya

dimana *Vd*, *Id* adalah titik operasi LED yang diinginkan, dan *VOL* adalah tegangan rendah output driver LED. Jika kita menggunakan nilai resistor standar 100 Ω menggantikan 90 Ω, maka arus menjadi , yaitu sekitar 9 mA. Arus rendah ini biasanya diterima.

Ketika arus LED kurang dari 8 mA, antarmuka secara langsung ke pin output tanpa menggunakan driver. LED yang ditunjukkan pada Gambar 6a memiliki titik operasi 1,6 V dan 1 mA.



Gambar 6. Antarmuka LED arus rendah

Untuk antarmuka logika positif (Gambar 6b) kita hitung nilai resistor berdasarkan pada tegangan dan arus LED yang diinginkan

dimana *VOH* adalah tegangan tinggi output pin output mikrokontroler. Berhubung *VOH* dapat bervariasi dari 2,4-3,3 V, masuk akal untuk memilih resistor dari nilai terukur *VOH*, daripada nilai minimum 2,4 V.

Logika negatif berarti LED diaktivasi ketika perangkat lunak output nilai nol. Untuk antarmuka logika negatif (Gambar 6c) kita menggunakan persamaan yang sama untuk menentukan nilai resistor

Dimana *VOL* adalah tegangan rendah output pin output mikrokontroler.

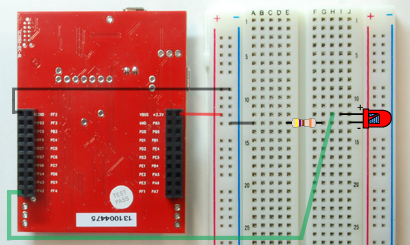
Jika kita menggunakan 1,2 kΩ menggantikan 1,3 kΩ, maka arus menjadi , yaitu sekitar 1,08 mA. Arus tinggi ini biasanya diterima. Jika kita menggunakan nilai resistor standar 1,5 kΩ menggantikan 1,3 kΩ, maka arus menjadi , yaitu sekitar 0,87 mA. Arus rendah ini biasanya diterima.

Perangkat lunak dalam Program 2 disebut **driver,** dan mencakup inisialisasi yang dipanggil sekali, dan dua fungsi yang dapat dipanggil untuk menghidupkan dan mematikan LED.

|  |
| --- |
| **void LED\_Init(void){ volatile unsigned long delay;   SYSCTL\_RCGC2\_R |= 0x01;           // 1) activate clock for Port A   delay = SYSCTL\_RCGC2\_R;           // allow time for clock to start                                     // 2) no need to unlock PA2   GPIO\_PORTA\_PCTL\_R &= ~0x00000F00; // 3) regular GPIO   GPIO\_PORTA\_AMSEL\_R &= ~0x04;      // 4) disable analog function on PA2   GPIO\_PORTA\_DIR\_R |= 0x04;         // 5) set direction to output   GPIO\_PORTA\_AFSEL\_R &= ~0x04;      // 6) regular port function   GPIO\_PORTA\_DEN\_R |= 0x04;         // 7) enable digital port } // Make PA2 high void LED\_On(void){   GPIO\_PORTA\_DATA\_R |= 0x04; } // Make PA2 low void LED\_Off(void){   GPIO\_PORTA\_DATA\_R &= ~0x04; }** |

Program 2. Antarmuka perangkat lunak untuk LED pada PF2 (C8\_LED)

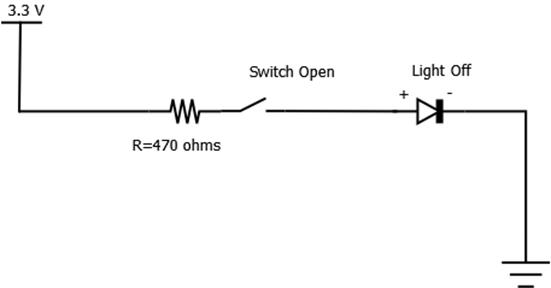
Penulisan perangkat lunak seperti ini disebut **abstraction, karena memisahkan apa yang LED lakukan (**Init, On, Off) dari cara kerjanya (PortA, bit 2, TM4C123).

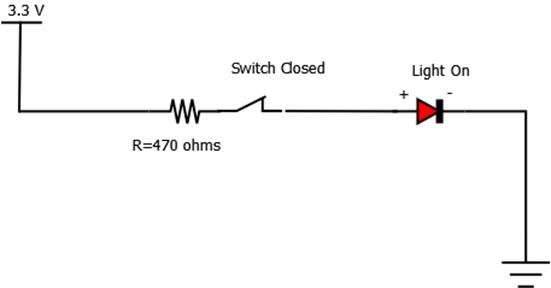


Gambar 7. Konstruksi antarmuka LED ke output mikrokomputer

1. **Eksperimen**

Gambar 8 memperlihatkan sistem logika positif.





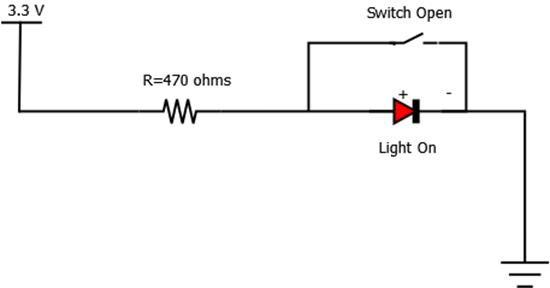
Gambar 8. Sistem logika positif

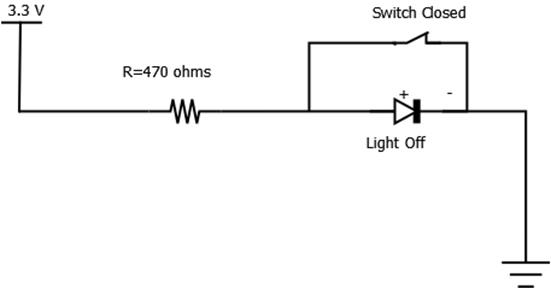
Jika saklar ditekan, arus mengalir dan LED hidup. Jika saklar tidak ditekan, arus tidak mengalir dan LED mati.

**Instruksi**:

Buat rangkaian pada protoboard berdasarkan Gambar 8, dan amati apa yang terjadi.

Gambar 9 memperlihatkan sistem logika negatif.





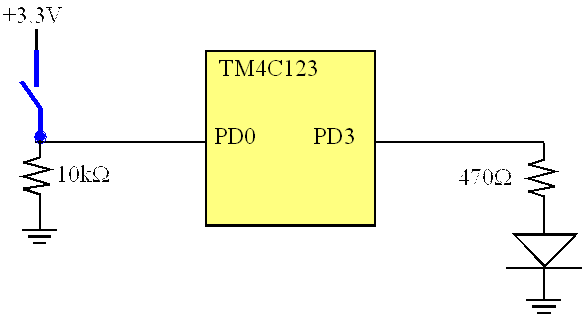
Gambar 9. Sistem logika negatif

Jika saklar tidak ditekan, arus mengalir melalui LED dan LED hidup. Jika saklar ditekan, seluruh arus mengalir melalui saklar, dan LED mati.

**Instruksi**:

Buat rangkaian pada protoboard berdasarkan Gambar 9, dan amati apa yang terjadi.

Gambar 10 memperlihatkan saklar logika positif dihubungkan ke PD0 dan LED logika positif dihubungkan ke PD3.



Gambar 10. Rangkaian mikrokontroler dengan saklar dan LED

Program 3 menginisialisasi PD0 sebagai input dan PD3 sebagai output.

|  |
| --- |
| **unsigned long in,out; int main(void){ unsigned long volatile delay;   SYSCTL\_RCGC2\_R |= 0x08;           // Port D clock   delay = SYSCTL\_RCGC2\_R;           // wait 3-5 bus cycles   GPIO\_PORTD\_DIR\_R |= 0x08;         // PD3 output   GPIO\_PORTD\_DIR\_R &= ~0x01;        // PD0 input    GPIO\_PORTD\_AFSEL\_R &= ~0x09;      // not alternative   GPIO\_PORTD\_AMSEL\_R &= ~0x09;      // no analog   GPIO\_PORTD\_PCTL\_R &= ~0x0000F00F; // bits for PD3, PD0   GPIO\_PORTD\_DEN\_R |= 0x09;         // enable PD3, PD0   while(1){     in = (GPIO\_PORTD\_DATA\_R&0x01); // in 0 if not pressed, 1 if pressed     out = (in^0x01)<<3;   // out 8 if not pressed, 0 if switch pressed     GPIO\_PORTD\_DATA\_R = out;   } }** |

Program 3. Sistem perangkat lunak yang menghidupkan LED jika saklar tidak ditekan

Loop utama melakukan fungsi yang diinginkan: jika saklar tidak ditekan maka LED hidup.

**Instruksi**:

* Buat rangkaian pada protoboard berdasarkan Gambar 10.
* Jalankan C:\Keil\TExaSware\C8\_SwitchLED\C8.uvproj.
* Periksa kesesuaian Program 3 dengan C8.uvproj.
* Gunakan simulator, amati apa yang terjadi.
* Gunakan Stellaris ICDI, amati apa yang terjadi.

1. **Bibliografi**

<http://users.ece.utexas.edu/~valvano/Volume1/E-Book/C8_SwitchLED.htm>