Modul 1: DIGITAL INPUT/OUTPUT

**Muhammad Syahreza Ishak 13318043, Kelompok :19**

Tanggal: 31 Januari 2020 Assisten: Krisna Diastama 13317016

TF 2207 - Laboratorium Teknik Fisika II

# Tujuan Percobaan (Heading 1)

1. Menjelaskan program pencacah 1 digit
2. Menjelaskan program *stopwatch 3 digit*
3. Menjelaskan program *keypad*  untuk kalkulator

.

# Alat dan Bahan

Table 2.1 Nama Tabel (gunakan References->Insert Caption)

|  |  |  |  |
| --- | --- | --- | --- |
| No | Nama | Banyak | Keterangan |
| 1  2  3  4  5  6  7 | Komputer / Laptop  Arduino + Kabel USB  Kit Arduino  Kit 7 segment  Kit 7 keypad  Power Supply  Kabel IDC -10 | 1  1  1  1  1  1  2 |  |

# Teori Dasar

Mikrokontroller arduino dapat menerima dan mengeluarkan sinyal listrik melalui pin input/outpit. Pin input/output ini dibagi menjadi dua jenis berdasarkan sinyal listriknya, yaitu analog dan digital. Sinyal analog berupa mapping voltase 0 sampai nilai maksimal Vin yang digunakan menuju nilai 0 sampai 1023[[1]](#footnote-1). Sementara itu, sinyal digital memiliki voltase 0-0,8 volt untuk nilai LOW, dan 4,75-5,25 volt untuk nilai HIGH. Dalam pemakaiannya, pin analog juga bias digunakan sebagai pin digital.

Pada mikrokontroller jenis Arduino uno, terdapat enam pin yang dapat digunakan untuk input sinyal analog (pin A0-A5) dan 14 pin untuk input/output digital (pin 0-13)[[2]](#footnote-2). Tidak ada pin output analog pada arduino ini. Output ini diganti dengan sinyal PWM dengan nilai 0 sampai 255[[3]](#footnote-3) pada pin digital 3, 5, 6, 9, 10, dan 11.

Jumlah pin arduino yang sedikit menyebabkan pengguna perlu menghemat jumlah pin yang digunakan. Penghematan ini dapat dulakukan dengan menggunakan perangkat keras decoder dan encoder. Decoder berperan menerjemahkan output dari mikrokontroller menjadi output seharusnya menuju perangkat keras yang ingin digunakan. Encoder berperan menerjemahkan input dari perangkat keras yang digunakan menuju mikrokontroller.

Decoder pertama yang digunakan pada praktikum ini adalah decoder BCD to 7 Segment 74LS47. Decoder ini menerima empat masukan biner yang menyatakan suatu angka desimal

0-15 dan mengeluarkan tujuh output menuju 7 segment agar 7 segment menampilkan angka desimal tersebut. Decoder selanjutnya adalah 7402N yang berisi empat NOR gate yang pada praktikum ini disusun agar menerima dua input yang menyatakan angka biner dan menghasilkan empat output untuk menyalakan tiga transistor ditambah satu titik di digit ketiga.

Encoder yang digunakan pada praktikum ini adalah chip 74C922. Chip ini menerima masukan dari 4 baris dan 4 kolom pada keypad dan mengeluarkan empat output yang bilangan desimal dalam bentuk biner beserta satu output yang menunjukkan tertekan atau tidak tombol pada keypad.

# Data dan Analisis

## Pencacah 1 Digit

**4.1.1 Program Seven Segment – Counter**

Hasil yang kita dapatkan mengunnakan seven segmant dengan software arduino sesuai denga data sheet IC 74LS47 dan yang sudah diberikan di modul 1 ini, sebab tampilan angka 1-15 dengan menggunakan seven-segment IC 74LS47 ini sesuai yang sudah ditetapkan dalam data sheetnya. Buktinya sudah terlmpir pada jurnal modul 1 ini. Tingakt kecerahan LED yang digunakan untuk menunjukkan angka 1-15 tidak mengalami perubahan dalam segi kecerahan dan dinilai “cerah” saat menunjukkan angak 1 sampai dengan 15 dan berjalan secara lancar dan konstan. Ketika dibandingkan dengan kelompok 20, terdapat banyak kesamaan diantara hasil percobaan satu ini dengan kelompok ini, salah satu tingkat kecerahan angka 1 sampai dengan 15 dinyatakan cerah dan berjalan secara konstan dan terang dari awal sampai akhir percobaan.

Fungsi resistor sendiri pada rangkaian elektronika sendiri adalah sebagai penahan tegangan dan arus, tentu apabila kita menghubungkan LED langsung denga \n power tanpa resistor makan LED nya akan hancur. Jika resistor tersambung dengan anpda pada 7-segmen *common anode* ini, resistor menerima arus gabungan dari ketujuh LEDnya, sehingga resistor lebih cepat panas atau akan rusak apabila melebihi daya maksimumnya.

**4.1.2 Program Counter- State Driven**

Program ini bertujuan untuk menghitung naik maupun turun suatu angka dan menampilkannya menuju 7-segmen. Pembuatan syntax pada pemrograman percobaan ini berbasis state driven yang memiliki arti alur utama program berdasarkan sifat dan keadaan yang dimilikinya pada saat itu dan hanya bisa berjalan apabila memuhi sifat dan kondsis pada masing-masing baris syntax. Kemudian untuk menjelaskan lebih lanjut ada dua state dalam program ini, state up dan state down. Saat state up, counter bertambah jika S1 ditekan dan state berubah menjadi state down jika S2 ditekan. Saat state down, program utama mengikuti alur S1 mengurangi counter dan S2 dapat mengganti kembali ke state up. Program yang kami buat berhasil bekerja seperti ini dan hasilnya terdapat pada lampiran.

**4.1.3 Program Counter – Event Driven**

Percobaan selanjutnya yang akan kami melakukan sebenarnya memiliki tujuan dan alur pemrograman yang mirip dengan percobaan sebelumnya, namun yang berbeda hanya bahwa percobaan berbasis event-driven, dimana konsep kerjanya tergantung dari kejadian atau event tertentu, misalnya ketika tombol A diklik makan nilai label 2 ditambah nilai label 3 dibagi nilai label 4. Yang paling berbeda event driven dengan state driven yaitu konsep pemilihan event driven untuk mengeksekusi proses programnya. Pada program ini, event yang saat tombol S1 atau S2 ditekan. Event penekanan S1 akan mengakibatkan aksi menambah atau mengurangi counter berdasarkan state program. Sedangkan event penekanan S2 akan mengakibatkan aksi perubahan state menjadi state up atau state down. Progam event driven yang kami buat juga dapat bekerja dengan benar.

* 1. **Stop- Watch 3 Digit**
     1. **Program Multiplexing**

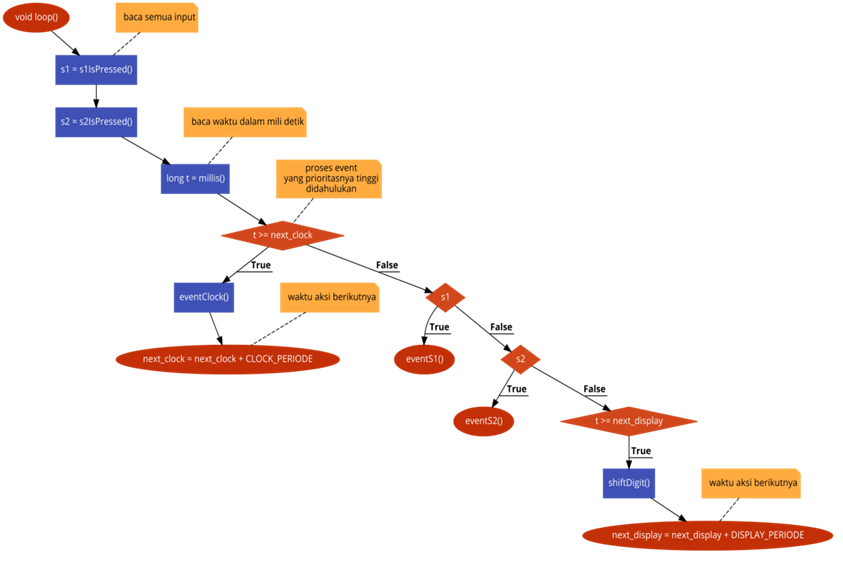
Multipeksing adalah teknik mengirimkan banyak informasi melalui satu saluran.[[4]](#footnote-4) Program ini bertujuan untuk menampilkan tiga digit angka secara berurutan di tiga 7-segmen. Dalam program ini, multipleksing dilakukan dengan menyatukan saluran luaran program menuju tiga 7-segmen menjadi satu. Hal ini dilakukan oleh subrutin shiftDigit(). Subrutin ini menginstruksikan chip 7402N agar menyalakan 7-segmen sesuai dengan yang diinginkan.

Subrutin shiftDigit() dijalankan setiap periode tertentu. Periode ini menjadi periode kilasan cahaya pada 7-segmen. Saat awal, kami mengatur periode kilasan sebesar satu detik, teramati 7-segmen berkedip. Periode kilasan diturunkan menjadi 100, 50, 22, dan 10 milidetik dengan kondisi 7-segmen masih berkedip. Saat periode diganti menjadi 7 milidetik, ketiga 7-segmen terlihat menyala secara terus-menerus. Oleh karena itu, display periode optimal yang kami peroleh adalah milidetik dimana cahaya yang dikeluarkan oleh LED-nya bersifat konstan dan tidak berkedip-kedip di padangan manusia. Perubahan kilasan yang masih dapat diamati manusia yaitu sekitar 80Hz atau periode 12.5 milidetik, tetapi dapat berubah bergantung beberapa faktor.[[5]](#footnote-5)

**4.2.2. Program Stopwatch**

Program ini bertujuan menayangkan waktu yang dihitung dari saat tombol S1 ditekan dan menghentikan penghitungan saat tombol ini ditekan kedua kalinya, kemudian melanjutkannya saat ditekan kembali, dan seterusnya. Saat tombol S2 ditekan, penghitungan akan berulang dari nol. Karena pada syntax pemrograman ini kita menggunakan loop maka siklusnya akan berjalan terus sampai ada fungsi stop ataupun delay. Waktu penghitungan akan ditampilkan pada ketiga 7-segmen dengan periode display optimal dan syntax permrograman pada percobaan program sebelumnya. Penambahan waktu terkecil pada program dapat diatur dengan konstanta CLOCK\_PERIOD dalam milidetik agar jarak waktu antar tiap angka terlihat lebih jelas di mata manusia.

Program ini berbasis event-driven dengan empat event utama yaitu saat S1 ditekan, S2 ditekan, periode penambahan clock terjadi, dan periode display terjadi. Berikut gambar flowchart dari subrutin loop dari program ini:

****

Gambar 1 Flowchart subrutin loop pada program Stopwatch.

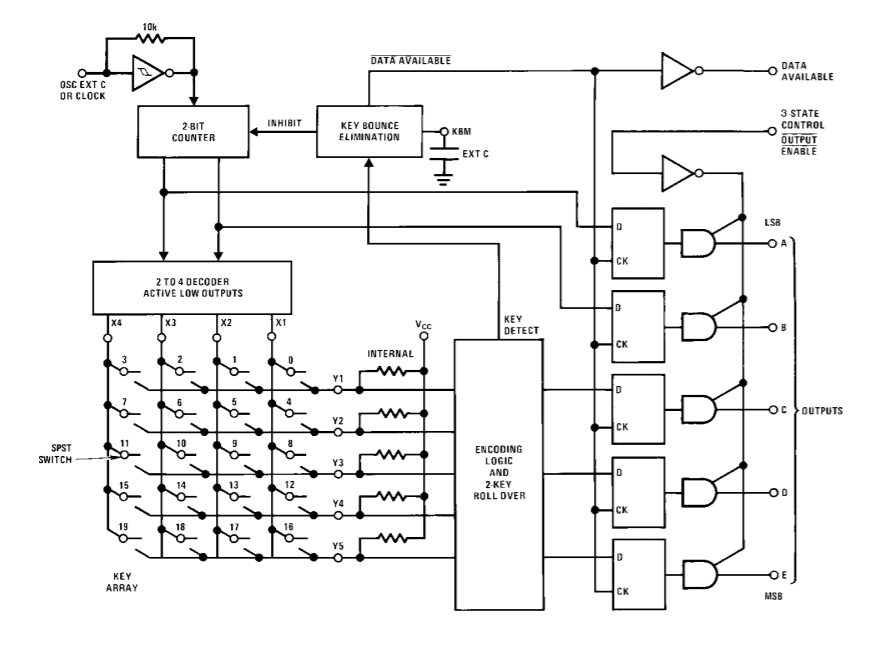
* 1. **Keypad untuk Kalkulator**

**4.3.1. Program Keypad**

Program ini bertujuan untuk membaca masukan dari keypad pada kit dan menampilkannya pada 7-segmen. Pembacaan ini dilakukan dengan mengamati empat masukan dari luaran chip 74\*922 berdasarkan keadaan tombol-tombol keypad. Pada percobaab ini kita akan memanfaatkan fungsi dan sifat encoder yang dapet menerjemahkan dan mengkonversi sinyal DATA\_AV dari LOW menjadi HIGH. Selain itu,masukan yang kita menginout ke dala perangkat lunaknya Arduinonya akan kemudian diterjemahkan oleh fungsi readKey() dengan diubah menjadi integer dari 0 sampai 15. Integer ini selanjutnya dijadikan indeks dari array biner\_table[], berisi nilai tombol yang ditekan. Setelah mengurutkan indeks berdasarkan nilai yang sebenarnya, kami memperoleh array biner\_table[] sebagai berikut:

|  |
| --- |
| char biner\_table[] = {0xD, 0xE, 0, 0xF, 0xC, 9, 8, 7, 0xB, 6, 5, 4, 0xA, 3, 2, 1}; |

Berdasarkan *datasheet* chip 74\*922, berikut adalah blok diagram chip ini untuk 20 tombol keypad:



Gambar 2 Blok diagram chip 74\*922 untuk 20-key (16-key dapat menyesuaikan)

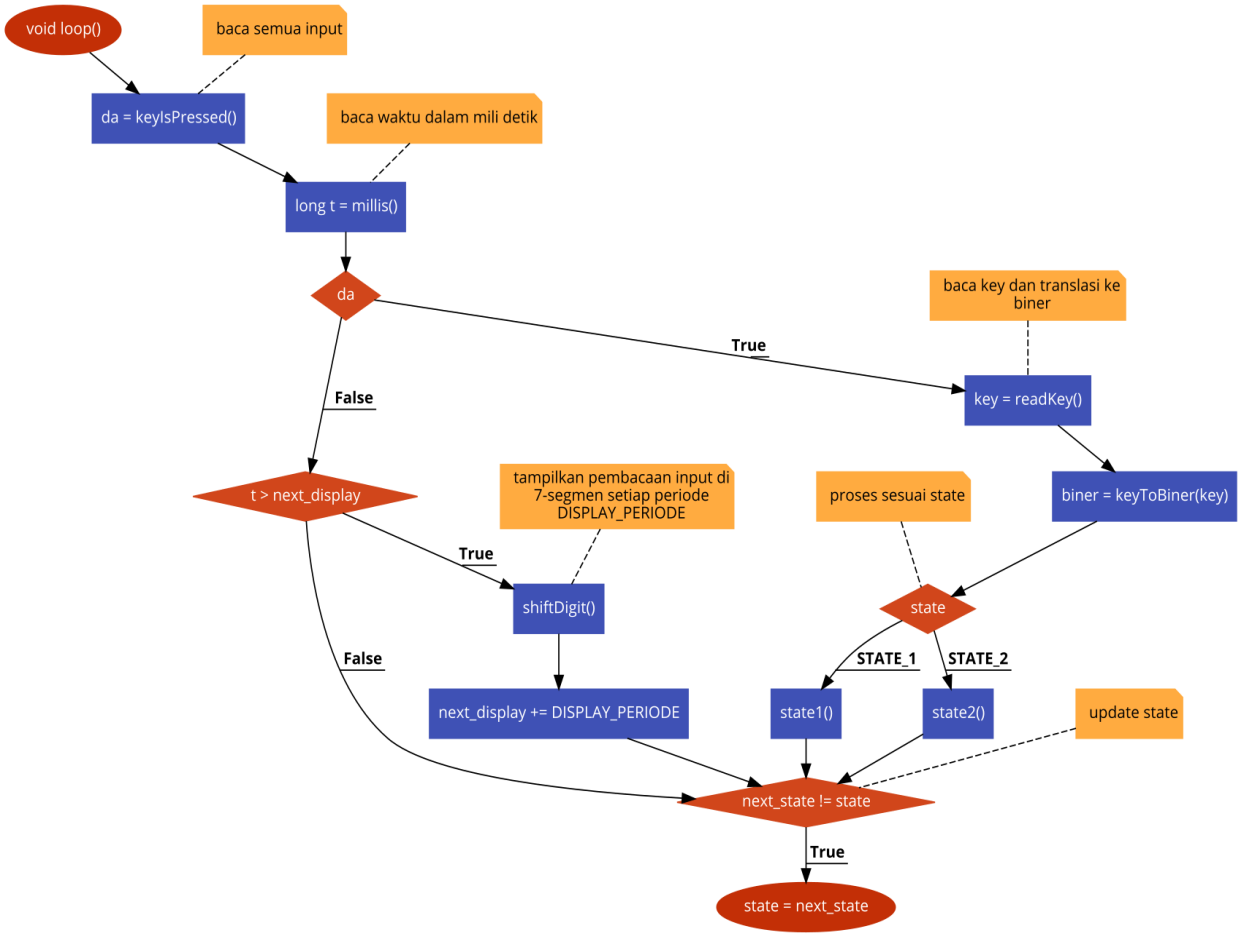
Terlihat bahwa jika tombol 1 *keypad* berada pada *key array* 15 dan tombol D berada pada key array 0 (tidak ada gambar *keypad* dari kit), maka indeks dari array biner\_table[] yang kami temukan telah sesuai dengan datasheet chip ini.

**4.3.2 Program Kalkulator**

Program ini bertujuan untuk mengubah keypad pada kit menjadi sebuah kalkulator sederhana. Program ini berbasis *state-driven.* Terdapat 2 state, state 1 merupakan kondisi awal saat angka pertama dimasukkan, angka ini dapat dimasukkan satu-persatu seperti memasukkan angka pada kalkulator biasa. Jika pengguna salah memasukkan angka, tombol bintang (\*) dapat ditekan dan angka kembali menjadi nol. Setelah angka berhasil dimasukkan, operator dapat dipilih dengan menekan tombol A, B, C, atau D dengan urutan operasi yaitu penjumlahan, pengurangan, perkalian, dan pembagian. Setelah itu, state program akan berubah menjadi state 2.

Pada state 2, program menerima angka seperti state 1. Jika masukan salah, tombol bintang akan mengubah state ke state 1 sebelum ada input angka. Jika angka telah dimasukkan, tombol operasi kembali dapat diretekan untuk melanjutkan operasi dengan angka luaran dari operasi sebelumnya. Saat operasi selesai, tombol pagar (#) dapat ditekan untuk mengeluarkan hasil penghitungan state akan dikembalikan ke state 1.

Berikut flowchart dari subrutin loop pada program ini:



Gambar 3 Flowchart subrutin loop pada program Calculator.

Setelah melakukan percobaan ini, data yang dihasilkan oleh program syntax kami sesuai dengan data yang diharapkan di awal percobaan, tentu walaupan sedikit terhambat oleh kualitas perangkat keras Arduino dan perangkat kalkulator yang tidak terlalu bagus untuk digunakan sehingga terdapat beberapa kesalahan dalam pengamatan kami ini.

# Kesimpulan

1. Program pencach 1 digit menampilkan angka di 7-segmen yang ditambah atau dikurang jika tombol ditekan dengan bantuan chip 74LS47
2. Program stopwatch 3 digit yang dibuat merupakan program berbasis event-driven yang menayangkan waktu pada tiga 7-segmen dengan teknik multipeksing oleh fungsi shiftDigit () menuju chip 7402N
3. Program keypad untuk kalkulator yang dibuat merupakan program berbasis state-driven yang membaca tombol keypad melalui chip 74\*922 dan mengubahnya menjadi angka atau operasi hitung sederhana.

# Daftar Pustaka

Anon., n.d. analogRead()*.* [Online]   
Available at: <https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/>   
[Accessed 2 Februari 2020].

Anon., n.d. Compare board specs*.* [Online]   
Available at: <https://www.arduino.cc/en/products.compare>   
[Accessed 2 Februari 2020].

Anon., n.d. Multipleksing*.* [Online]   
Available at: <https://id.wikipedia.org/wiki/Multipleksing>   
[Accessed 3 Februari 2020].

Anon., n.d. Flicker fushion threshold*.* [Online]   
Available at: <https://en.wikipedia.org/wiki/Flicker_fusion_threshold>   
[Accessed 3 Februari 2020].

# Lampiran

Daftar File Terlampir:

1. analogRead(), <https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/> [↑](#footnote-ref-1)
2. Compare board specs, <https://www.arduino.cc/en/products.compare> [↑](#footnote-ref-2)
3. PWM, <https://www.arduino.cc/en/Tutorial/PWM> [↑](#footnote-ref-3)
4. Multipleksing, <https://id.wikipedia.org/wiki/Multipleksing> [↑](#footnote-ref-4)
5. Flicker fushion threshold, <https://en.wikipedia.org/wiki/Flicker_fusion_threshold> [↑](#footnote-ref-5)