BAB IV

PENGUJIAN DAN EVALUASI SISTEM

Pengujian dan evaluasi sistem pada tugas akhir ini meliputi perangkat keras elektonika dan perangkat lunak yang telah dibuat. Pengujian pada perangkat keras elektronika menggunakan program simulasi *Proteus ISIS 7 Professional* sedangkan pengujian pada perangkat lunak dilakukan dengan menggunakan program simulasi *ProView32 Version 3.3* (*Franklin Sftware Inc*).

Dari pengujian ini akan didapatkan data-data maupun bukti-bukti bahwa sistem yang telah dibuat dapat bekerja dengan baik. Berdasarkan data-data dan bukti-bukti tersebut, akan dapat dilakukan analisa terhadap proses kerja, yang nantinya dapat digunakan untuk menarik kesimpulan dari apa yang telah dijelaskan dalam tugas akhir ini.

4.1. Pengujian Perangkat Keras Elektronika

Pengujian perangkat keras elektronika ini meliputi (modul mikrokontroler, komunikasi antar mikrokontroler, rangkaian driver motor MOC3021 dan rangkaian driver motor L298).

4.1.1 Modul Mikrokontroler

A. Tujuan

Tujuan dari pengujian rangkaian mikrokontroler ini adalah untuk mengetahui apakah rangkaian yang dibuat dapat berfungsi dengan baik. Dimana

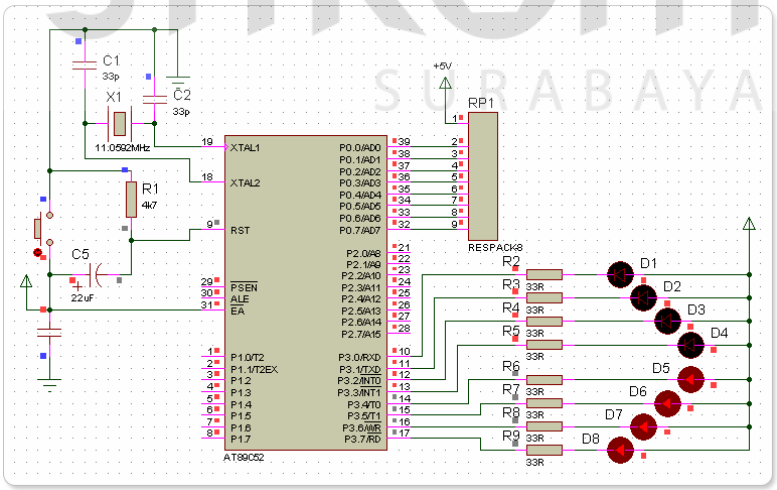
rangkaian mikrokontroler ini digunakan untuk mengatur dan mengendalikan semua proses produksi dan pengemasan sambal pecel.

B. Peralatan yang Digunakan

1. Program simulasi *Proteus ISIS 7 Professional.*
2. Rangkaian minimum sistem AT89C52 (proteus tidak menyediakan mikrokontroler AT89S52).
3. Catu daya 5 Volt.
4. Rangkaian LED sebanyak tuju buah (sebagai *display* atau output simulasi).

C. Prosedur Pengujian

1. Minimum sistem dan LED dirancang seperti pada Gambar 4.1. berikut. Ke tuju buah LED di hubungkan pada *port* P3 (*pin* P3.0-P3.7).



Gambar 4.1. Rangkaian Pengujian Minimum Sistem

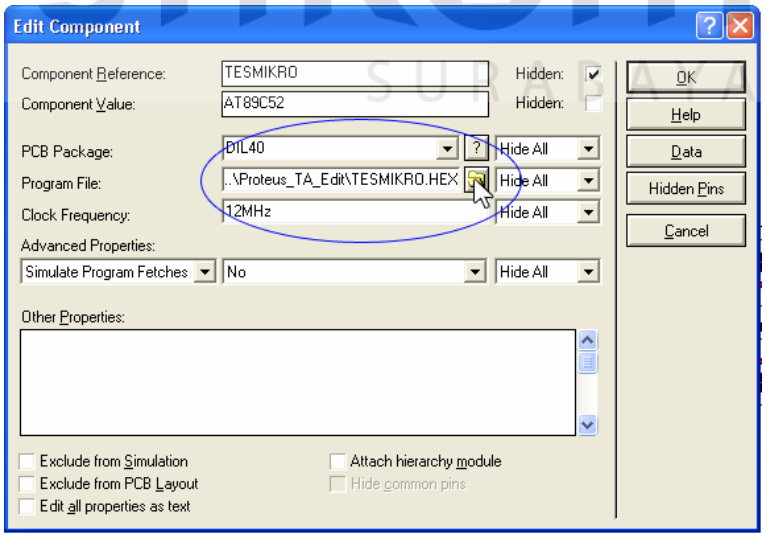
2. Membuat program sederhana untuk menyalakan lampu LED secara berjalan dengan menggunakan program ProView32. Setelah program dibuat

kemudian dilakukan debuger untuk membuat file.hex yang nantinya akan dimasukkan kedalam AT89C52.

Program sederhana pengujian minimum sistem:

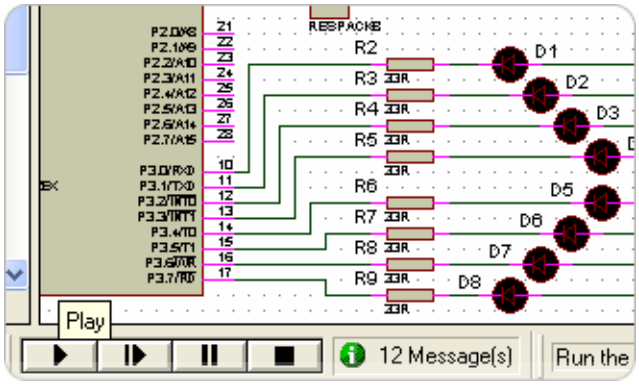
```
; file name: tesmikro.asm

ORG     000H

        MOV     A, #11111110B

MULAI:  MOV     P3,     A
        ACALL   DELAY
        RL      A
        SJMP    MULAI

DELAY:  MOV     R0,     #10H
DELAY1: MOV     R1,     #0FFH
DELAY2: MOV     R2,     #0H
        DJNZ    R2,     $
        DJNZ    R1,     DELAY2
        DJNZ    R0,     DELAY1
        RET
END
```

3. File.hex (tesmikro.hex) dimasukkan kedalam mikrokontroler AT89C52 dengan cara *double click* pada IC AT89C52.



Gambar 4.2. Cara Memasukkan File.hex Kedalam IC AT89C52

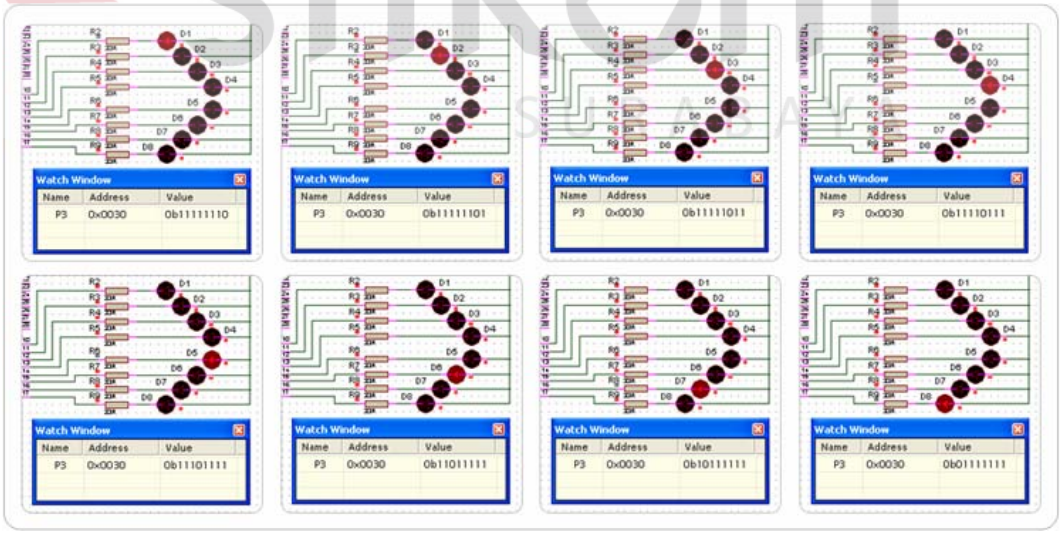
4. Tekan tombol play pada menu window untuk memulai simulasi dan lihat perubahan yang terjadi pada ke 7 LED (*display*).



Gambar 4.3. Tombol Play Untuk Memulai Simulasi

D. Hasil Pengujian

Berikut ini adalah hasil pengujian dari modul mikrokontroler yang dilakukan dengan menggunakan program *proteus ISIS 7 professional*.



Gambar 4.4. Tampilan LED Menyala Secara Berjalan

Tabel 4.1 Hasil Pengujian Mikrokontroler

Output LED (Display)	Address	Value	Keterangan
Port P3 (*pin* P3.0-P3.7)	0x0030	0b 11111110 0b 11111101 0b 11111011 0b 11110111 0b 11101111 0b 11011111 0b 10111111 0b 01111111	Lampu LED menyala dari atas ke bawah atau dari kanan ke kiri (RL), kemudian kembali lagi dari awal sampai simulasi dihentikan (diakhiri).

E. Analisa

Dari hasil pengujian di atas dapat disimpulkan bahwa mikrokontroler AT89C52 dapat berfungsi dengan baik sesuai dengan perancangan yang dibuat.

4.1.2 Komunikasi Antar Mikrokontroler

A. Tujuan

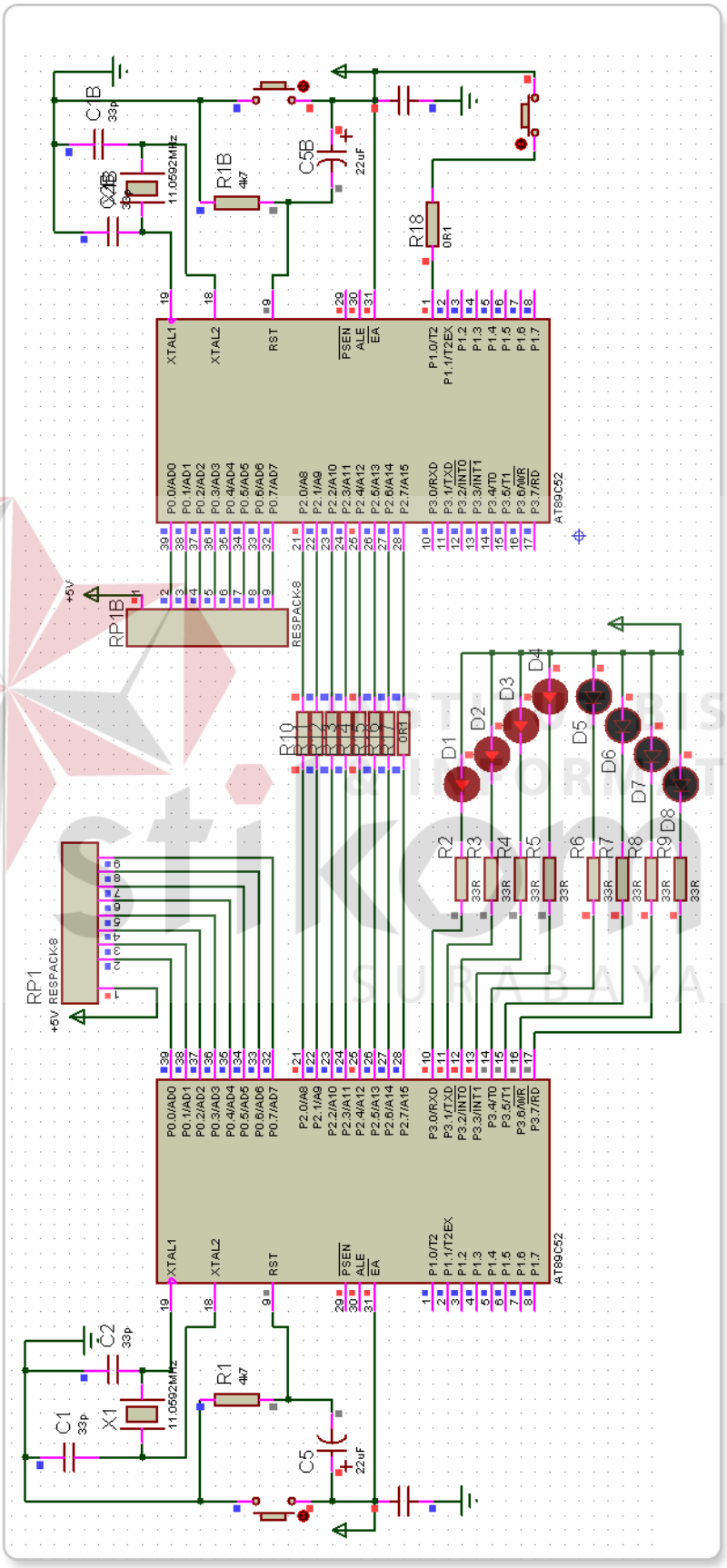
Tujuan dari pengujian komunikasi antar mikrokontroler ini adalah untuk mengetahui apakah komunikasi yang dilakukan antara mikrokontroler pertama dengan mikrokontroler kedua dapat berjalan dengan baik.

B. Peralatan yang Digunakan

1. Program simulasi *Proteus ISIS 7 Professional*.
2. Dua buah rangkaian minimum sistem AT89C52.
3. Tuju buah LED sebagai *disply* simulasi.
4. *Limit switch* sebagai inputan pengiriman data mikrokontroler kedua kepada mikrokontroler pertama.
5. Catu daya 5 Volt.

C. Prosedur Pengujian

1. Kedua buah rangkaian minimum sistem AT89C52 (mikrokontroler pertama dan kedua) serta ketuju buah LED dirangkai seperti pada Gambar 4.5 berikut. Komunikasi antar mikro menggunakan *port* P2 (*pin* P2.0-P2.7), inputan limit switch menggunakan *port* P1 (*pin* P1.0) sedangkan *output* simulasi (tuju buah LED) menggunakan *port* P3 (*pin* P3.0-P3.7).



Gambar 4.5. Rangkaian Komunikasi Antar Mikrokontroler.

2. Membuat program komunikasi sederhana pada kedua buah mikrokontroler dengan menggunakan program ProView32. Setelah program dibuat kemudian dilakukan debuger untuk membuat file.hex yang nantinya akan dimasukkan kedalam kedua buah AT89C52.

Program sederhana komunikasi antar mikrokontroler:

; file name: teskomunikasimikro-1.asm

```
ORG     000H

RESET_PORT :
        MOV     P0, #00H
        MOV     P1, #00H
        MOV     P2, #00H
        MOV     P3, #00H

MULAI:

        MOV     A,P2            ;terima data dari mikro 2
        LCALL   DELAY
        CJNE    A,#11H, MULAI

HIDUPKAN_LAMPU:
        MOV     P3,     #00001111B
        ACALL   DELAY
        MOV     P3,     #11110000B
        ACALL   DELAY
        SJMP    MULAI

DELAY: MOV      R0,     #5H
DELAY1:  MOV    R1,     #0FFH
DELAY2:  MOV    R2,     #0H
                DJNZ    R2,     $
                DJNZ    R1,     DELAY2
                DJNZ    R0,     DELAY1
                RET
END
```

; file name: teskomunikasimikro-2.asm

```
ORG     000H

SAKLAR          BIT     P1.0

RESET_PORT :
        MOV     P0, #00H
        MOV     P1, #00H
        MOV     P2, #00H
        MOV     P3, #00H
```

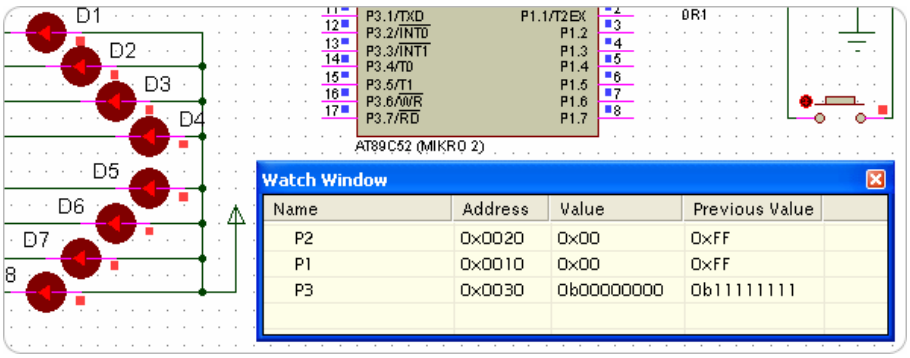
```
MULAI:

        JNB     SAKLAR,$
        MOV     A,#11H      ;kirim data 11h ke mikro 1
        MOV     P2,A
        LCALL   DELAY

DELAY:  MOV     R0,     #5H
DELAY1: MOV     R1,     #0FFH
DELAY2: MOV     R2,     #0H
                DJNZ    R2,     $
                DJNZ    R1,     DELAY2
                DJNZ    R0,     DELAY1
                RET
END
```

3. File.hex (teskomunikasimikro-1.hex dan teskomunikasimikro-2.hex) dimasukkan kedalam masing-masing mikrokontroler AT89C52 dengan cara seperti pada Gambar 4.2. prosedur pengujian modul mikrokontroler di atas.
4. Tekan tombol play untuk memulai simulasi. Seperti pada Gambar 4.3. prosedur pengujian modul mikrokontroler di atas.

D. Hasil Pengujian

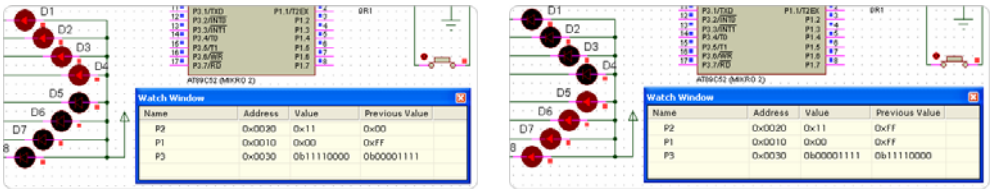
Berikut ini adalah hasil pengujian komunikasi antar mikrokontroler yang dilakukan dengan menggunakan program *proteus ISIS 7 professional*.

- Kondisi sebelum terjadinya komunikasi (*limit switch* pada mikro 2 *pin* P1.0 belum ditekan atau mikro 2 belum mengirim data #11H ke mikro 1).



Gambar 4.6. Sebelum Terjadi Komunikasi.

- Kondisi Setelah terjadi komunikasi (*limit switch* pada mikro 2 *pin* P1.0 ditekan atau mikro 2 mengirim data #11H ke mikro 1). LED 4-bit pertama dan LED 4-bit kedua akan menyala dan mati secara bergantian.



Gambar 4.7. Setelah Terjadi Komunikasi

Tabel 4.2. Hasil Pengujian Komunikasi Antar Mikrokontroler

Limit switch	Input (Mikro 2)	Value (Output Mikro 1)	Keterangan
Off	0x00	0b 11110000	LED 4-bit pertama (*pin* P3.0-P3.3) menyala dan LED 4-bit kedua (*pin* P3.4-P3.7) mati.
On	0x11	0b 00001111 0b 11110000 0b 00001111 0b 11110000 0b 00001111	LED 4-bit pertama dan LED 4-bit kedua menyala dan mati secara bergantian. Hal ini akan terus dilakukan sampai simulasi dihentikan atau limit switch dimatikan.

E. Analisa

Dari hasil pengujian komunikasi antar mikrokontroler, dapat ditarik kesimpulan bahwa *output*, *input*, dan komunikasi kedua mikrokontroler dapat berjalan dengan baik sesuai dengan perancangan sistem.

4.1.3 Driver Motor MOC3021

A. Tujuan

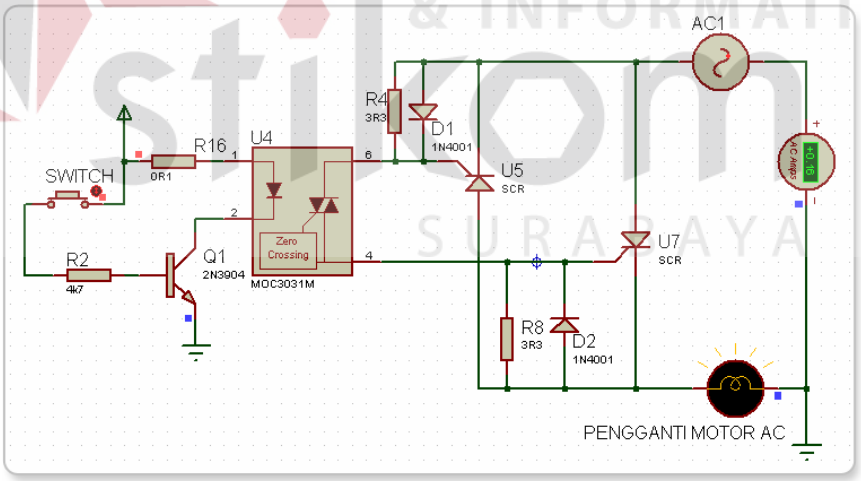
Tujuan dari pengujian driver MOC3021 ini adalah untuk mengetahui apakah rangkaian dapat mengendalikan motor AC dengan baik.

B. Peralatan yang Digunakan

1. Program simulasi *Proteus ISIS 7 Professional.*
2. Rangkaian driver MOC3021.
3. Lampu AC (*proteus* tidak menyediakan motor AC)
4. Limit switch sebagai masukan 5 Volt pengganti inputan dari mikrokontroler
5. Catu daya 5 Volt.
6. Catu daya 220 Volt.

C. Prosedur Pengujian

1. Driver motor MOC3021 dan lampu AC dirangkai seperti pada Gambar 4.8. berikut.



Gambar 4.8. Rangkaian Driver Motor MOC3021

2. Tekan tombol play pada menu window *proteus* untuk memulai simulasi.
3. Tekan *limit switch* untuk menghidupkan lampu (inputan 5 Volt sebagai masukan IC MOC3021).

D. Hasil Pengujian

Tabel 4.3. Hasil Pengujian Driver Motor MOC3021

Limit switch (inputan 5 Volt)	Lampu AC pengganti motor AC
Pasif	Mati
Aktif	Menyala

E. Analisa

Dari hasil pengujian driver motor MOC3021 dapat ditarik kesimpulan bahwa rangkaian dapat berjalan dengan baik sesuai dengan rancangan.

4.1.4 Driver Motor L298

A. Tujuan

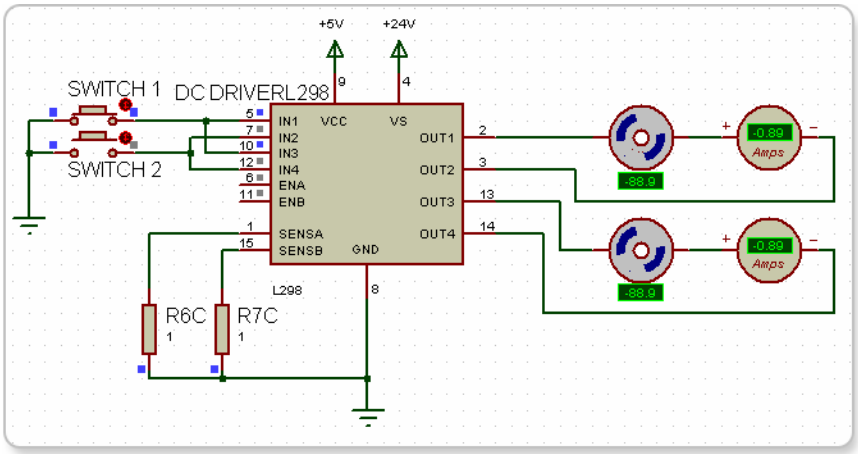
Tujuan dari pengujian ini adalah untuk mengetahui apakah rangkaian driver L298 dapat mengendalikan motor DC dengan baiak.

B. Peralatan yang Digunakan

1. Program simulasi *Proteus ISIS 7 Professional.*
2. Rangkaian driver L298.
3. Dua buah motor DC.
4. Dua buah *limit switch*.
5. Catu daya 5 Volt

C. Prosedur Pengujian

1. Driver L298, motor DC dan *limit switch* di rangkai seperti pada Gambar 4.9. berikut.



Gambar 4.9. Rangkaian Driver Motor L298

2. Tekan tombol play pada menu window *proteus* untuk memulai simulasi.

3. Tekan *limit switch* untuk mengatur putaran motor (*limit switch* 1= putaran motor berlawanan dengan arah jarum jam dan *limit switc* 2= putaran motor searah dengan arah jarum jam).

D. Hasil Pengujian

Tabel 4.4. Hasil Pengujian Driver Motor L298

Limit switch (1)	Limit switch (2)	Respon Motor	Arah Putaran Motor
Pasif	Pasif	Diam	-
Aktif	Aktif	Diam	-
Aktif	Pasif	Berputar	Berlawanan arah jarum jam
Pasif	Aktif	Berputar	Searah jarum jam

E. Analisa

Dari hasil pengujian terhadap rangkaian driver motor L298 di atas, maka dapat ditarik kesimpulan bahwa rangkaian dapat mengendalikan motor DC dengan baiak.

4.2. Pengujian Perangkat Lunak

A. Tujuan

Tujuan dari pengujian ini adalah untuk mengetahui apakah program (perangkat lunak) yang telah dibuat dapat menangani *input* dan *output* sesuai dengan cara kerja dari sistem yang di buat.

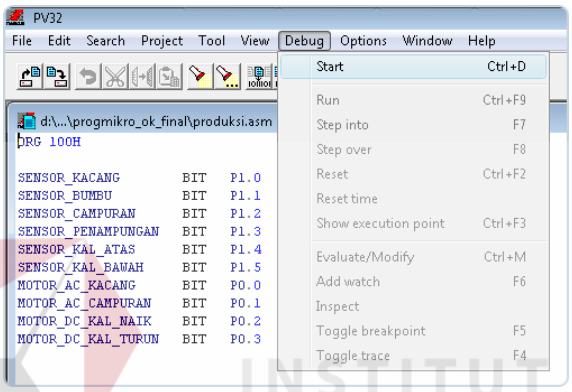
B. Peralatan

Peralatan yang dibutuhkan untuk melakukan pengujian terhadap sistem ini adalah: “Sebuah PC yang digunakan untuk menjalankan program simulasi serta program simulasi ProView32 yang digunakan untuk melakukan simulasi dari program *asembly* yang dibuat pada mikrokontroler pertama dan mikrokontroler kedua.

C. Prosedur Pengujian

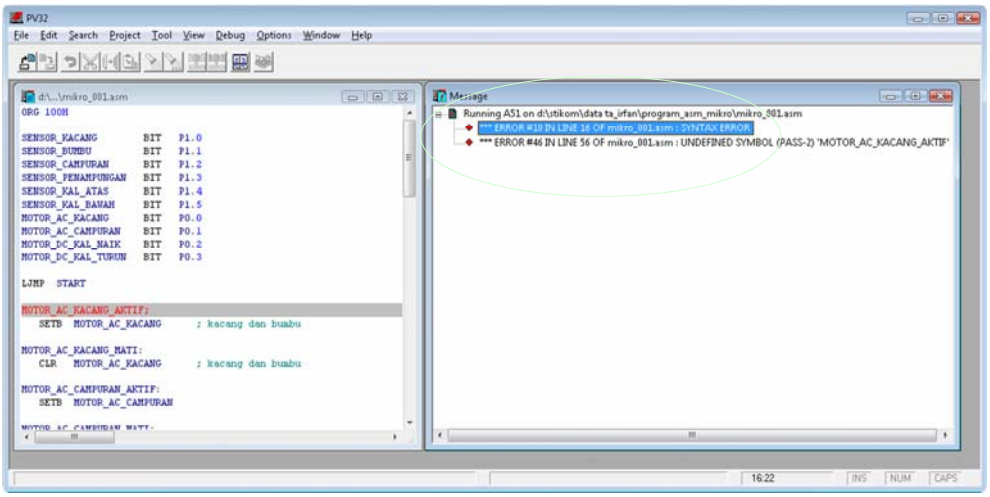
Program mikrokontroler pertama dan mikrokontroler kedua dibuat dengan menggunakan bahasa *asembly* melalui program ProView32, kedua buah mikrokontroler diprogram secara terpisah dengan nama dan fungi yang berbeda. Program mikrokontroler pertama disimpan dengan nama file (*Produksi.asm*) yang diprogram untuk menangani proses-proses pada peralatan produksi sambal pecel, sedangkan mikrokontroler kedua disimpan dengan nama file (*Pengemasan.asm*) yang diprogram untuk menanagani semua proses yang terjadi pada mesin pengemasan sambal pecel.

Setelah program selesai dibuat, langkah selanjutnya adalah menjalankan program untuk mengetahui sukses dan tidaknya terhadap program yang telah kita buat. Caranya adalah, pada *keyboard* tekan tombol [Ctrl] + [D] atau pada menu window pilih Debag kemudian pilih Start, Gammbar 4.10. Menu *ProView32* untuk melakukan *debag*.



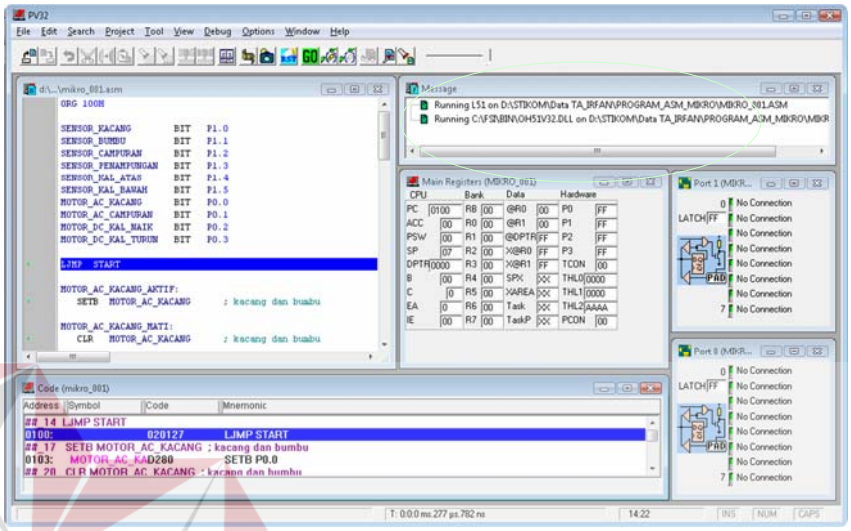
Gambar 4.10. Menu ProView32

Bila dalam pembuatan program ini terdapat kesalahan dalam penulisan *syntax*, prosedur atu yang lain maka pada window *message* akan ditampilkan pesan kesalahan-kesalahan dari program yang kita buat, tampilan pesan kesalahan dapat dilihat pada Gambar 4.11. berikut:



Gambar 4.11. Pesan Kesalahan Program

Setelah program dilakukan pembetulan dari kesalahan-kesalahan yang ada, maka tampilan dari ProView adalah seperti Gambar 4.12. berikut.



Gambar 4.12. Tampilan Program Yang Sudah Benar

D. Hasil Pengujian

Untuk mengetahui hasil pengujian dari program yang telah kita buat, kita dapat melihat perubahan nilai yang terdapat pada *main register* atau perubahan warna indikator *view hardware* pada masing-masing port.

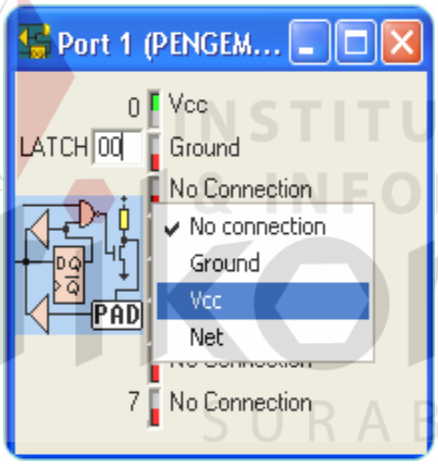
Dalam simulasi ini perubahan warna idikator pada *view hardware* mempunyai arti sama terhadap semua *input* dan *output*, warna hijau menunjukkan suatu keadaan aktif (hidup) sedangkan warna merah menunjukkan bahwa *pin* yang bersangkutan dalam kondisi pasif (mati).

Setelah program di jalankan (*debag*), kemudian kita tekan [F7] pada *keyboard* untuk mengamati perubahan yang terjadi secara *step by step* (langkah demi langkah) terhadap program yang dibuat.

D.1. Hasil Pengujian Perangkat Lunak Mikrokontroler Pertama

Sebagai pengganti *inputan* dari sensor ke mikrokontroler dapat dilakukan dengan cara, c*lick* kiri di kotak link pada *view hardware* untuk memilih salah satu dari empat pilihan, (*No connection, Ground, Vcc*, atau *Net*).

Jika ingin mengaktifkan sensor, maka link pada *view hardware* dipilih *Vcc* dan apabila ingin megembalikan kondisi semula (sensor yang sudah aktif ingin dinonaktifkan) maka link pada *view hardware* dipindah ke *Ground*, begitu seterusnya. Cara pengaturan (link) *view hardware* seperti yang terlihat pada Gambar 4.13.



Gambar 4.13. Pengaturan *View Hardware*

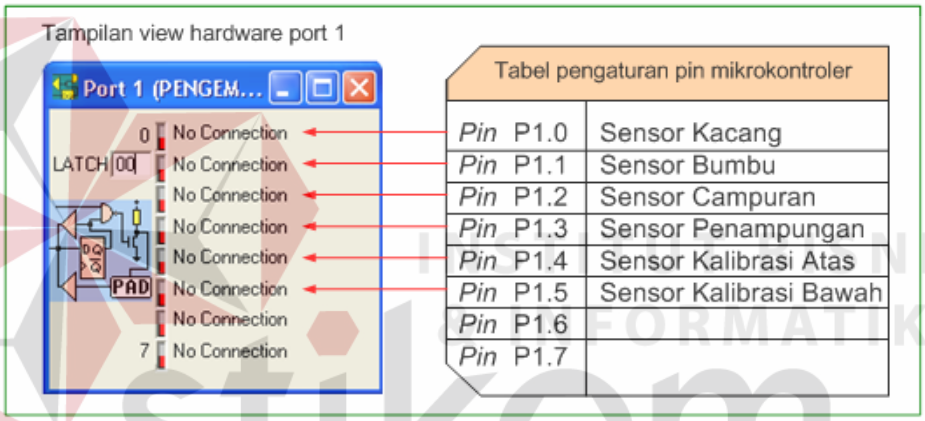
Untuk mengetahui fungsi-fungsi dan pemakaian *pin-pin* mikrokontroloer pertama tapat dilihat dalam Tabel 4.5. berikut.

Tabel 4.5. Pengaturan *Pin* Mikrokontroler Pertama

Nomor *Pin*	Fungsi	Keterangan
P1.0	Photodiode	Sensor Kacang
P1.1	Photodiode	Sensor Bumbu
P1.2	Photodiode	Sensor Campuran Kacang dan Bumbu
P1.3	Photodiode	Sensor Sambal pada Penampungan
P1.4	*Limit switch*	Sensor Batas Atas Kalibrasi

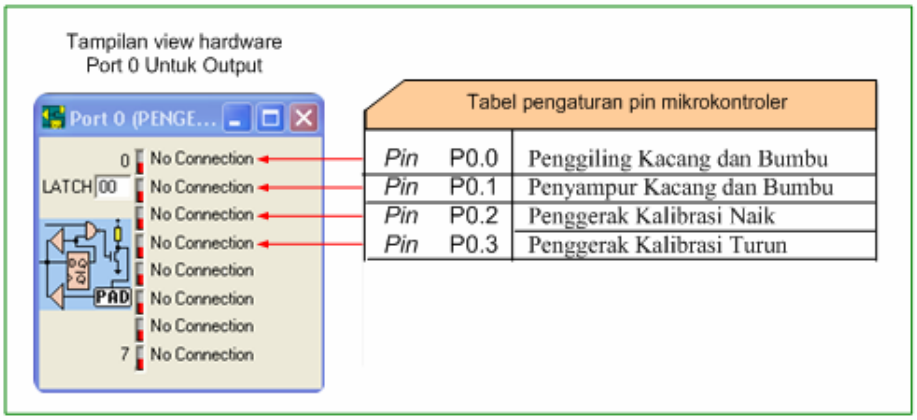
P1.5	*Limit switch*	Sensor Batas Bawah Kalibrasi
P0.0	Motor AC	Penggerak Gilingan Kacang dan Bumbu
P0.1	Motor AC	Penggerak Campuran Kacang dan Bumbu
P0.2-P0.3	Motor DC	Penggerak Naik Turun Kalibrasi
P2.0-P2.7	Koneksi	Komunikasi dengan Mikrokontroler Kedua

Seperti yang terlihat dalam tabel *Port* 1 (*pin* P1.0-*pin* P1.5) digunakan sebagai *port input* untuk sensor, Gammbar 4.14 berikut adalah tampilan ProView32 pengaturan *Port* 1 untuk *inputan*.



Gambar 4.14. Keterangan View Hardware Sebagai *Inputan*

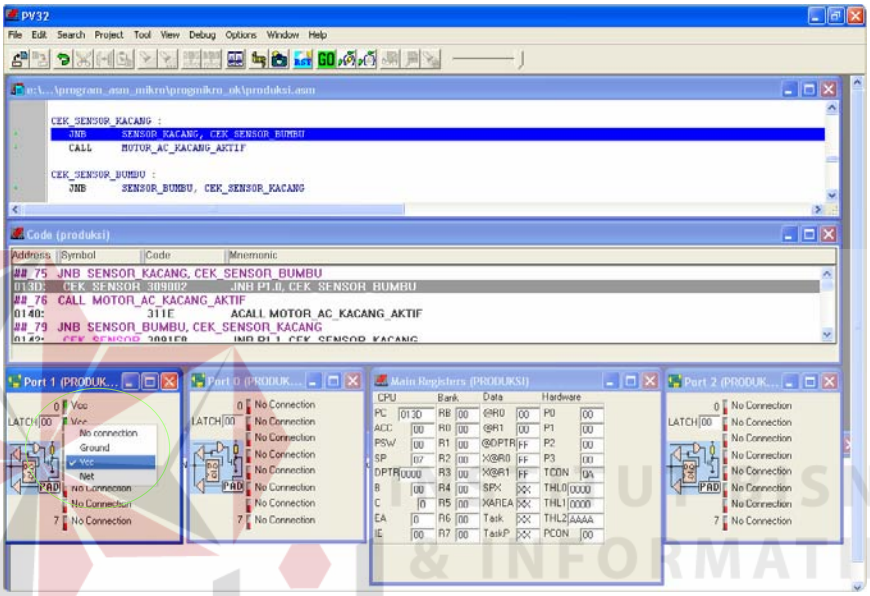
Sedangkan *Port* 0 (*pin* P0.0-*pin* P0.3) digunakan sebagai *port output* untuk motor AC dan motor DC. Pengaturannya seperti Gambar 4.15. berikut.



Gambar 4.15. Keterangan View Hardware Sebagai *Outputan*

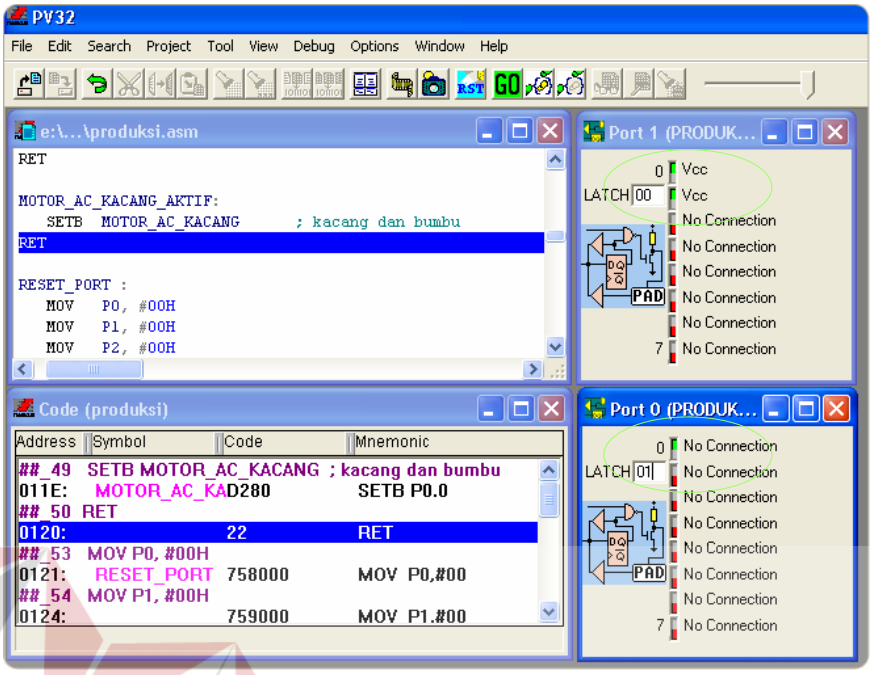
Hasil dari pengujian program mikrokontroler pertama yang dilakukan secara *step by step* diuraikan dibawah ini:

- *Pin* P1.0 dan *Pin* P1.1 dihubungkan ke Vcc (sensor kacang dan bumbu aktif). Gambar 4.16. Tampilan ProView32 sensor kacang dan bumbu aktif.



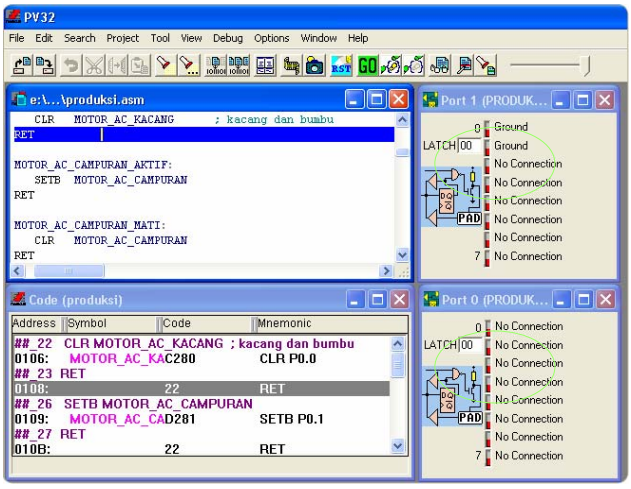
Gambar 4.16. Sensor Kacang dan Bumbu Aktif

- Aktifnya sensor kacang dan bumbu mengakibatkan *Pin* P0.0 motor AC aktif. Gambar 4.17. Proses penggilingan kacang dan bumbu.



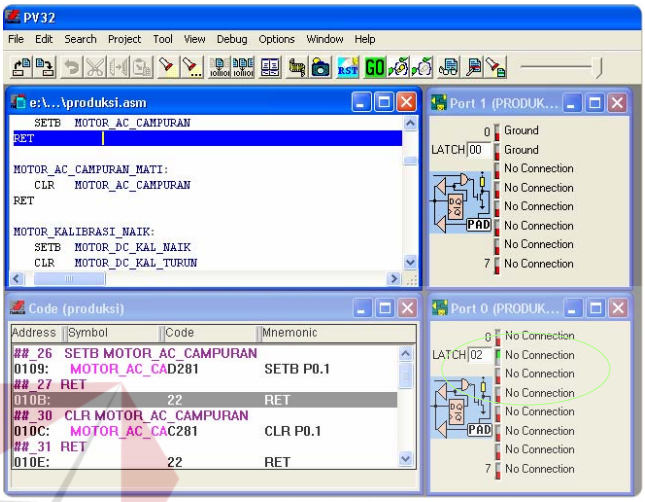
Gambar 4.17. Penggilingan Kacang dan Bumbu

- *Pin* P0.0 akan mati (penggilingan kacang dan bumbu selesai), apabila *Pin* P1.0 dan *Pin* P1.1 mati (tidak ada kacang dan bumbu yang terdeteksi oleh sensor), *link Pin* P1.0 dan *Pin* P1.1di *ground* kan. Keadaan ini ditunjukkan pada Gambar 4.18.



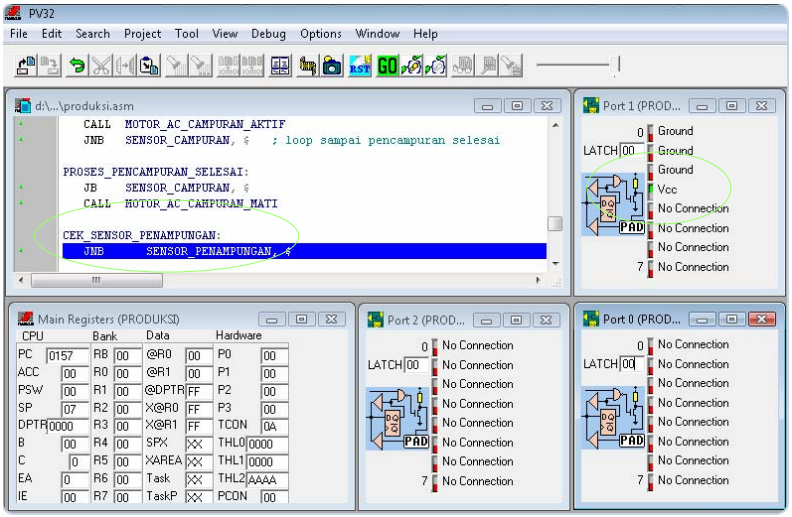
Gambar 4.18. Penggilingan Kacang dan Bumbu Selesai

- Matinya *Pin* P1.0 dan *Pin* P1.1 digunakan sebagai logika untuk menghidupkan *Pin* P0.1 (proses penyampuran kacang dan bumbu). Gambar 4.19. Tampilan ProView32 proses penyampuran kacang dan bumbu.



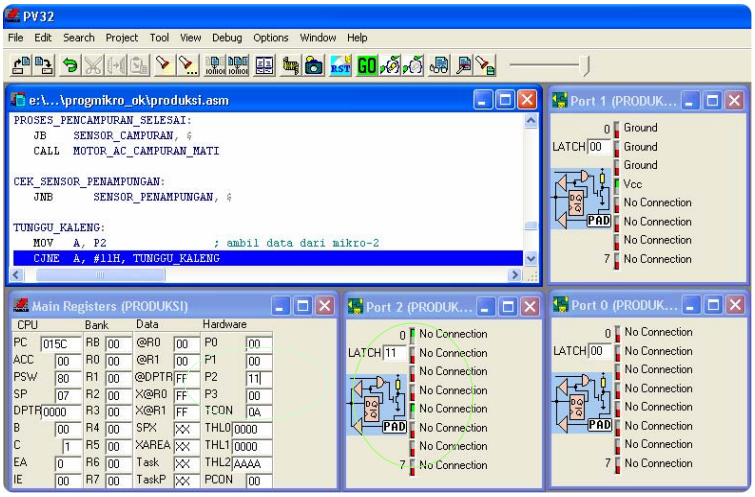
Gambar 4.19. Proses Penyampuran

- *Pin* P1.2 adalah sensor penyampuran yang digunakan untuk mendeteksi hasil campuran (sambal) yang keluar dari proses penyampuran menuju ke tempat penampungan. *Pin* P1.2 akan menyala selama sambal masih keluar dari mesin penyampuran, begitu sebaliknya. Ketika *Pin* P1.2 mati maka program akan mematikan *Pin* P0.1 (proses penyampuran selesai).
- Sambal yang berada dalam tempat penampungan akan dideteksi oleh sensor penampungan melalui *Pin* P1.3 (sensor penampungan aktif). Sensor ini sangat penting mengingat proses pengemasan akan terus dilakukan selama *Pin* P1.3 ini masih aktif (masih ada sambal yang perlu dikemas). Gambar 4.20. Sensor penampungan mendeteksi sambal



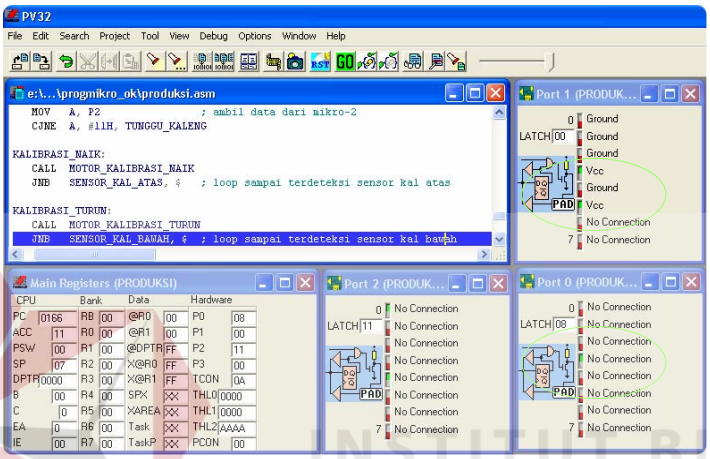
Gambar 4.20. Sensor Penampungan Aktif

- Sebelum dilakukan kalibrasi untuk proses pengisian, *port* 2 mikrokontroler pertama akan mengambil data dari mikrokontroler kedua (selama *port* 2 tidak bernilai 11H) maka proses pengisian tidak akan dilakukan, hal ini berarti kaleng yang berada pada proses pengemasan belum siap, begitu sebaliknya. Jika *port* 2 bernilai 11H maka proses kalibrasi akan dilakukan. Gambar 4.21. menunjukkan proses komunikasi dengan mikrokontroler kedua



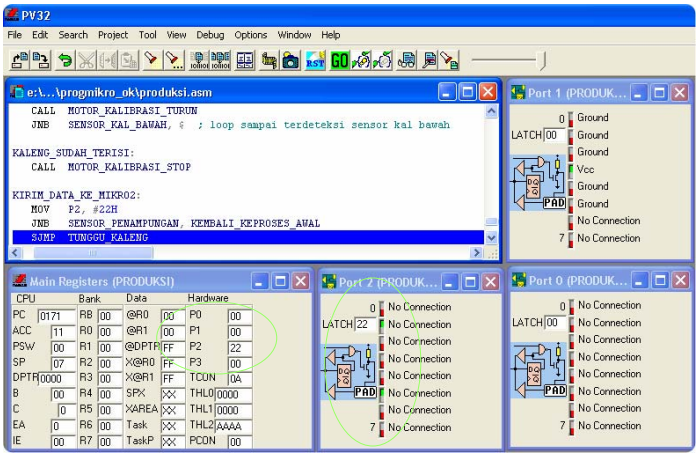
Gambar 4.21. Komunikasi Dengan Mikrokontroler Kedua

- *Pin* P0.2 (kalibrasi naik) akan aktif sampai menyentuh sensor atas kalibrasi (*pin* P1.4 aktif) lalu motor berhenti dan kemudian *pin* P0.3 (kalibrasi turun) akan aktif sampai sensor bawah kalibrasi *pin* P1.5 aktif motor berhenti. Gambar 4.22. Proses kalibrasi.



Gambar 4.22. Proses Kalibrasi

- Setelah proses kalibrasi diatas selesai, maka proses selanjutnya adalah mikrokontroler pertama mengirim data 22H melalui *port* 2 ke mikrokontroler kedua, sebagai informasi bahwa pengisian telah selesai proses ini ditunjukkan pada Gambar 4.23. berikut.



Gambar 4.23. Proses Pengisian Selesai Mikrokontroler Pertama Mengirim Data 22H melalui *port* 2 ke Mikrokontroler kedua

- Setelah proses ini selesai maka program telah berejalan satukali alur proses. Selanjutnya program akan mengecek sensor penampungan apakah masih ada sambal yang perlu dikemas bila tidak maka proses akan selesai dan bila masih ada sambal dalam penampungan maka proses akan kembali ke proses yang menunjuk pada Gambar 4.20.

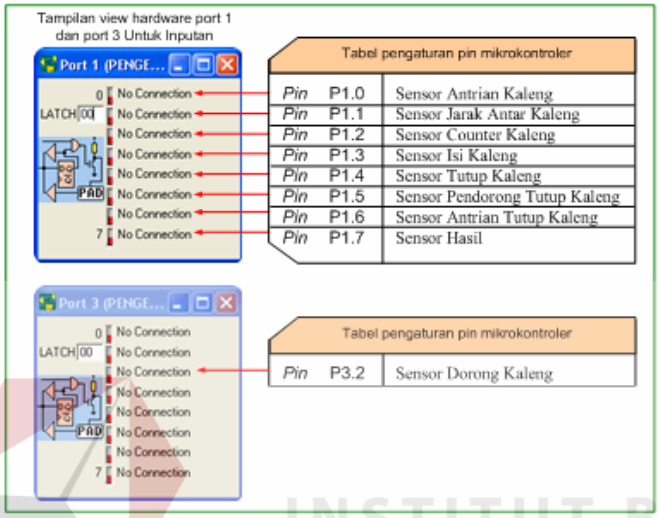
D.2. Hasil Pengujian Perangkat Lunak Mikrokontroler kedua

Pengujian yang dilakukan pada mikrokontroler kedua ini sama dengan pengujian yang dilakukan pada mikrokontroler pertama. Untuk mengetahui fungsi masing-masing *pin* pada mikrokontroler kedua dapat dilihat pada Tabel 4.6. berikut.

Tabel 4.6. Pengaturan *Pin* Mikrokontroler Kedua

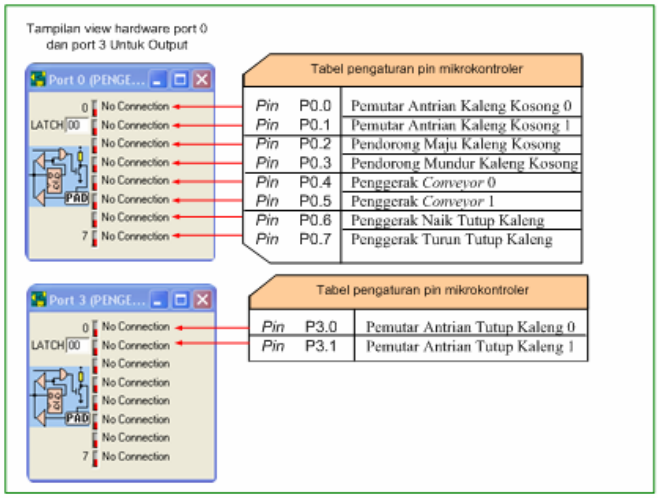
Nomor *Pin*	Fungsi	Keterangan
P1.0	Photodiode	Sensor Antrian Kaleng
P1.1	Photodiode	Sensor Jarak Antar Kaleng
P1.2	Photodiode	Sensor Counter Kaleng
P1.3	Photodiode	Sensor Isi Kaleng
P1.4	*Limit switch*	Sensor Tutup Kaleng
P1.5	*Limit switch*	Sensor Pendorong Tutup Kaleng
P1.6	*Limit switch*	Sensor Antrian Tutup Kaleng
P1.7	*Limit switch*	Sensor Hasil
P3.2	*Limit switch*	Sensor Dorong Kaleng
P0.0-P0.1	Motor DC	Pemutar Antrian Kaleng Kosong
P0.2-P0.3	Motor DC	Pendorong Maju Mundur Kaleng Kosong
P0.4-P0.5	Motor DC	Penggerak *Conveyor*
P0.6-P0.7	Motor DC	Penggerak Naik Turun Tutup Kaleng
P3.0-P3.1	Motor DC	Pemutar Antrian Tutup Kaleng
P2.0-P2.7	Koneksi	Komunikasi dengan Mikrokontroler Pertama

Seperti yang terlihat dalam Tabel 4.2. *Port* 1 (*pin* P1.0-*pin* P1.7) dan *Port* 3 (*pin* P3.2)digunakan sebagai *port input* untuk sensor, Gammbar 4.24 berikut adalah tampilan ProView32 pengaturan *Port* 1 dan *Port* 3 untuk *inputan* sensor.



Gambar 4.24. Keterangan View Hardware Untuk Sensor

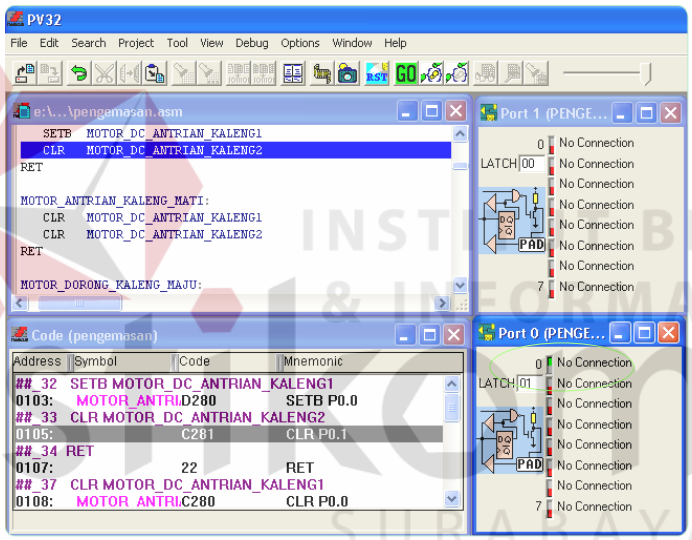
Port 0 (*pin* P0.0-*pin* P0.7) dan *Port* 3 (*pin* P3.0, *pin* P3.1) difungsikan sebagai port *output* dari motor DC. Pengaturan port *outputan* untuk motor dapat dilihat pada Gambar 4.25.



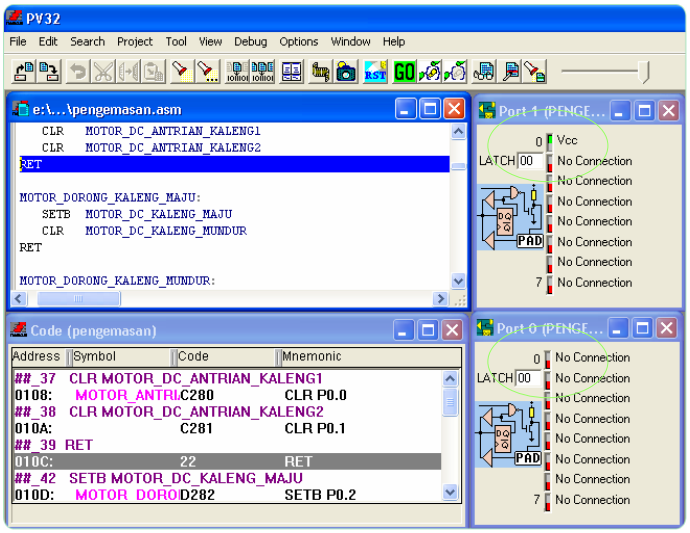
Gambar 4.25. Keterangan View Hardware Untuk Motor

Hasil pengujian program mikrokontroler kedu yang dilakukan secara *step by step* diuraikan sebagai berikut.

- Pada saat mikrokontroler kedua aktif program pertamakali yang dijalankan adalah mengeset *Pin* P0.0 (memutar antrian kaleng), dan akan berhenti sampai *Pin* P1.0 aktif (terdeteksi sensor antrian kaleng). Gambar 4.26. dan Gambar 4.27. berikut ini menunjukkan proses antrian kaleng berputar sampai terdeteksi sensor dan berhenti.

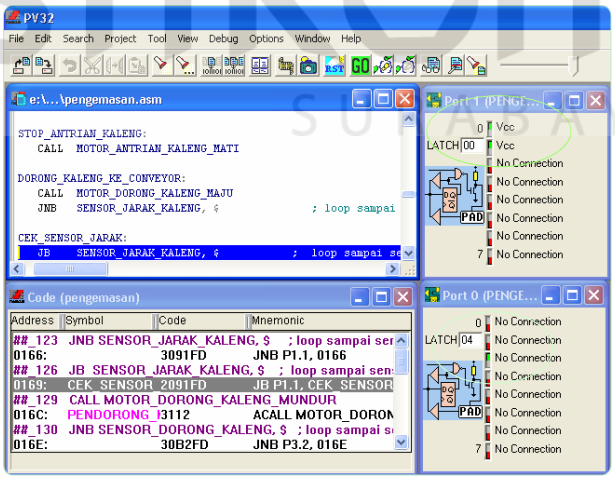


Gambar 4.26. Antrian Kaleng Berputar



Gambar 4.27. Antrian Kaleng Berhenti

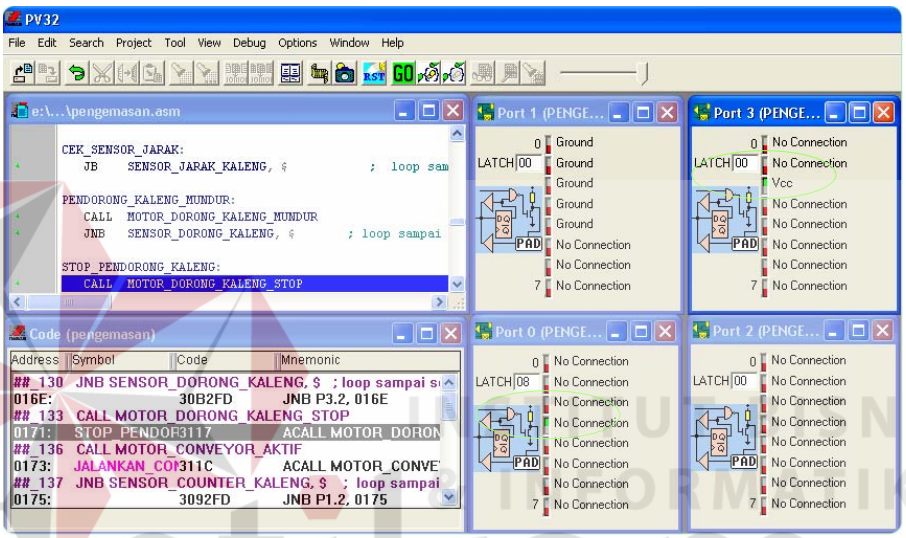
- Setelah *Pin* P0.0 (antrian kaleng) mati dan *Pin* P1.0 (sensor antrian kaleng) aktif, selanjutnya *Pin* P0.2 (dorong kaleng aktif) bergerak maju sampai *Pin* P1.1 (sensor jarak) aktif, kemudian berhenti. Gambar 4.28. proses dorong kaleng ke *conveyor*.



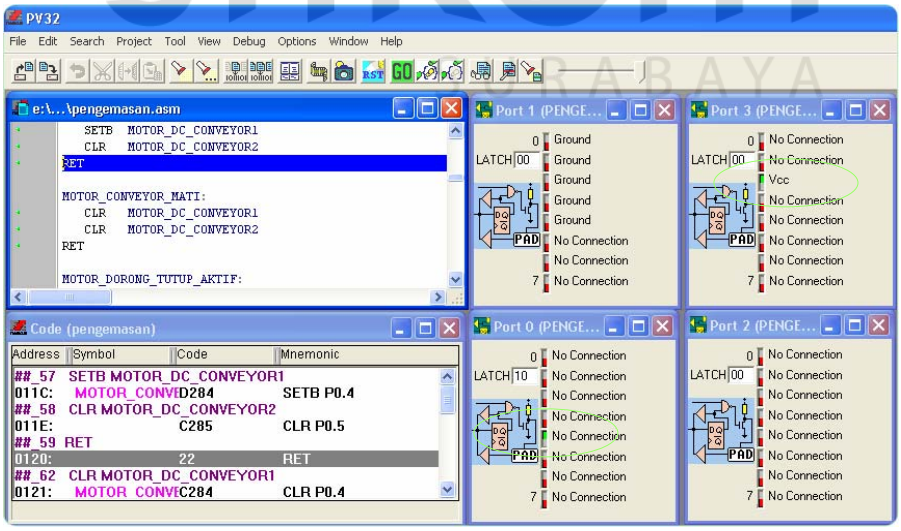
Gambar 4.28. Dorong kaleng ke *conveyor*

- *Pin* P1.1 (sensor jarak) aktif *Pin* P0.2 (dorong kaleng maju) mati dan kemudian *Pin* P0.3 (dorong kaleng mundur) bergerak ke belakang sampai

Pin P3.2 (sensor dorong kaleng) aktif, *Pin* P3.2 aktif menyebabkan *Pin* P0.3 mati dan *Pin* P0.4. (*conveyor* bergerak) aktif. Gambar 4.29. dan Gambar 4.30. berikut adalah proses yang menunjukkan pendorong kembali keposisi semula dan *conveyor* bergerak membawa kaleng kesatu.

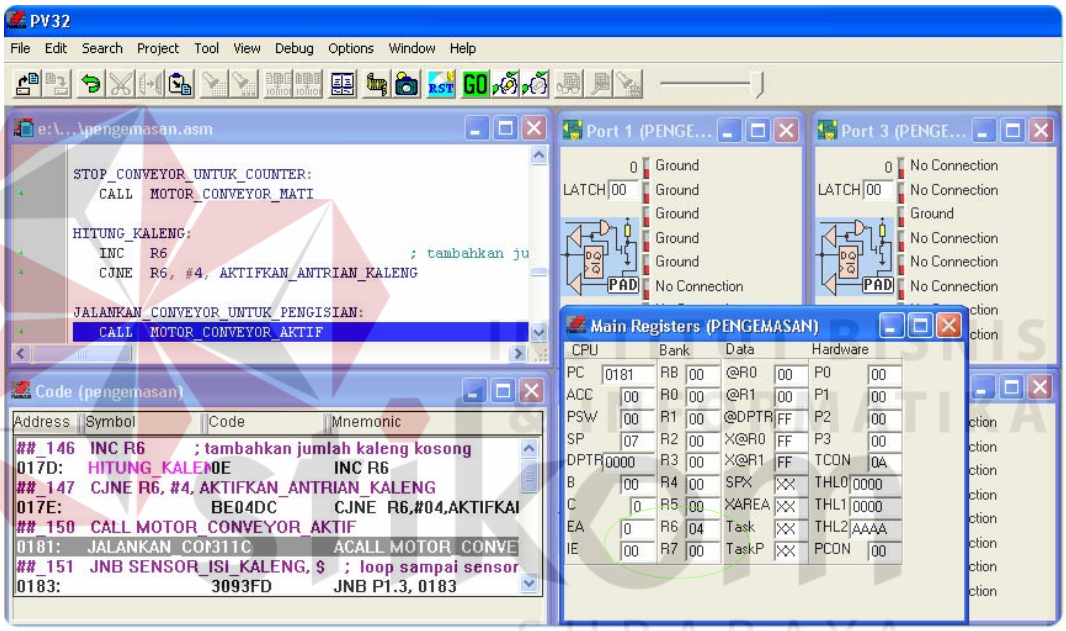


Gambar 4.29. Pendorong Bergerak Kebelakang



Gambar 4.30. Pendorong Berhenti dan *Conveyor* Bergerak

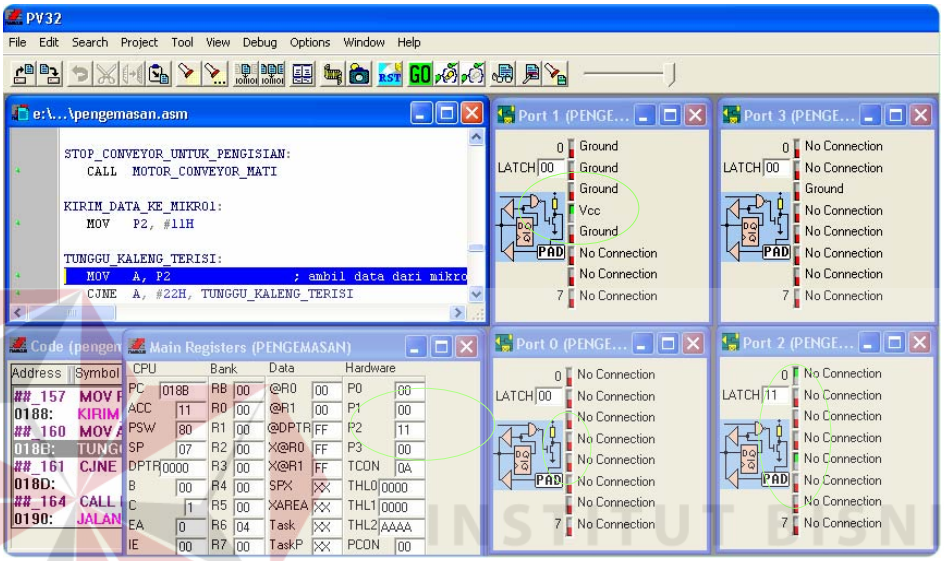
- *Pin* P0.4 (*conveyor*) bergerak sampai *Pin* P1.2 (sensor counter) aktif dan mati kemudian *conveyor* berhenti. Setelah berhenti *pin* P1.2 akan melakukan perhitungan naik pada R6 (register R6) sekaligus melakukan pengecekan apakah R6 sudah sama dengan 4, kalau belum maka proses seperti semula dilakukan lagi sampai nilai dari register (R6 = 4). Gambar 4.31. berrikut menunjukkan tentang proses perhitungan naik jumlah kaleng.



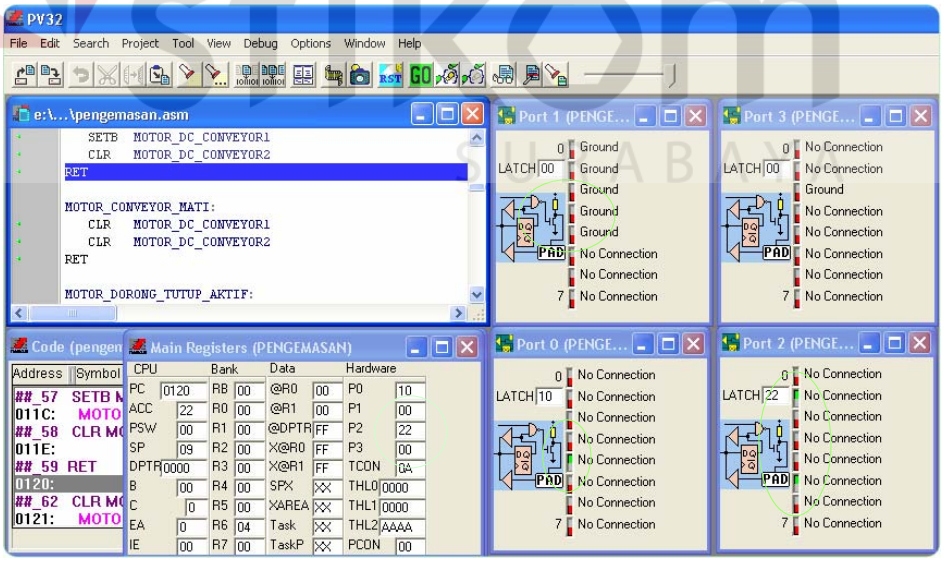
Gambar 4.31. Hitung Register (R6) Sampai Bernilai 4

- Bila R6 sama dengan 4, berarti sudah ada empat buah kaleng yang siap untuk dilakukan pengisian. *Conveyor* bergerak sampai *pin* P1.3 (sensor isi kaleng) aktif, ketika *pin* P1.3 aktif mikrokontroler kedua mengirim data 11H ke mikrokontroler pertama melalui *port* 2 untuk menginformasikan bahwa kaleng siap diisi (MOV P2, #11 H). Setelah kaleng terisi mikrokontroler pertama mengirim data 22H ke mikrolontroler kedua sebagai informasi pengisian telah selesai.

Kemudian *pin* P0.4 (*conveyor*) aktif untuk mebawa kaleng yang sudah terisi keproses penutupan kaleng sampai *pin* P1.4 (sensor tutup kaleng) aktif. Proses-proses tersebut dapat dilihat pada Gambar 4.32. dan Gambar 4.33.

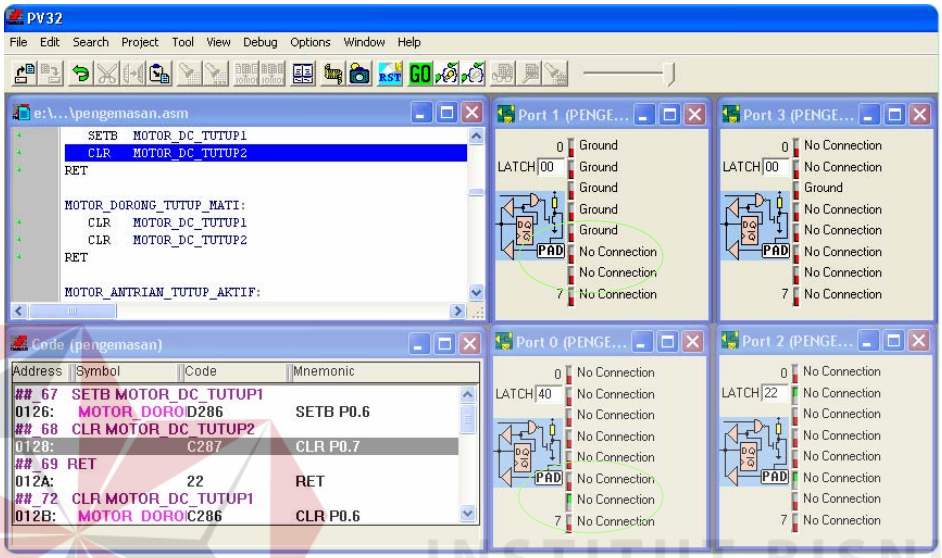


Gambar 4.32. Proses Pengisian Sambal Pecel



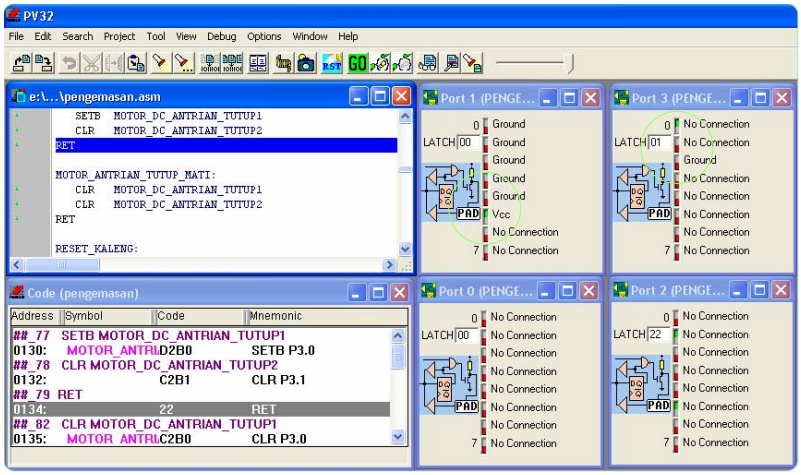
Gambar 4.33. Pengisian Selesai dan *Conveyor* Bergerak

- *Pin* P1.4 (sensor tutup kaleng) aktif *conveyor* tetap bergerak sampai *pin* P1.4 mati. Setelah *pin* P1.4 mati baru *conveyor* berhenti dan *pin* P0.6 (dorong tutup kaleng) akan aktif. Gambar 4.34. prooses penutupan kaleng.



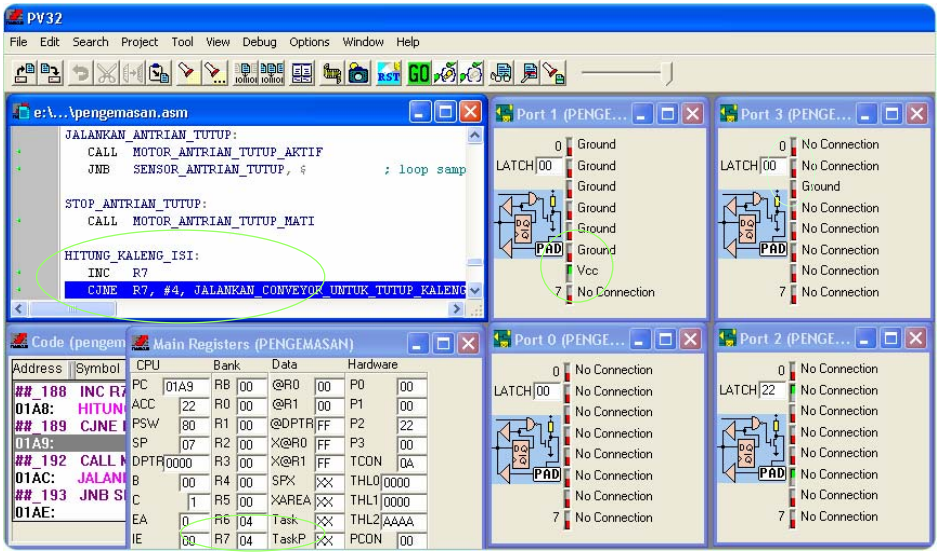
Gambar 4.34. Proses Penutupan Kaleng Pertama

- Ketika pendorong tutup kaleng mendorong tutup ke kaleng maka *pin* P1.5 (sensor pendorong tutup) akan aktif, aktifnya *Pin* P1.5 menyebabkan *Pin* P3.0 (motor antrian tutup kaleng) akan aktif (berputar 180 º) untuk menyiapkan tutup yang kedua. Putaran 180 º ini dihentikan atau diatur oleh sensor antrian tutup kaleng *Pin* P1.6 (sensor antrian tutup kaleng). Proses putaran antrian tutup kaleng dapat dilihat pada Gambar 4.35. berikut.



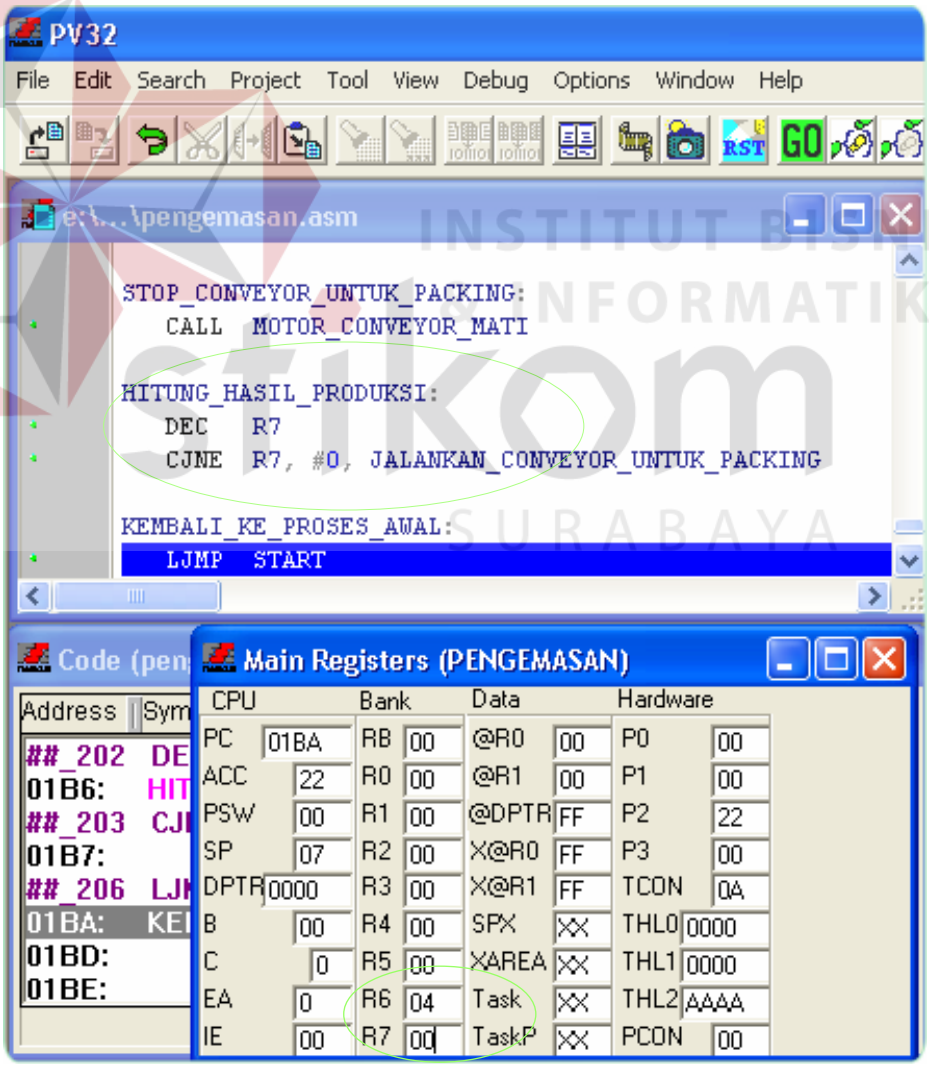
Gambar 4.35. Proses Antrian Tutup Kaleng

- Ketika *pin* P1.6 (sensor antrian tutup) aktif maka akan menghentikan *pin* P3.0 (motor antrian tutup kaleng) dan sekaligus akan melakukan perhitungan naik pada register R7 (INC R7) (CJNE R7, # 4) untuk mengetahui berapa kaleng yang sudah dilakukan penutupan. Jika R7 belum sama dengan 4, maka proses akan dilakukan seperti semula (program kembali ke proses yang menghasilkan Gambar 4.34.) sampai keempat kaleng trtutup semua. Proses perhitungan register R7 dapat dilihat pada Gambar 4.36. berikut.



Gambar 2.36. Proses Penghitungan Kaleng Yang Ditutup

- Proses terakhir Gambar 4.37. Setelah R7 sama dengan 4, maka selanjutnya keempat buah kaleng (yang sudah terisi sambal dan telah diberi tutup) dibawa *conveyor* untuk dilakukan penghitungan mundur (DEC R7) (CJNE R7,#0) oleh *Pin* P1.7 (sensor hasil) untuk benar-benar memastikan bahwa hasil produksi sama dengan kaleng yang disiapkan pada awal proses.



Gambar 4.37. Proses Terakhir Pengecekan Hasil Produksi