Tahmini Ders İçeriği
(Tentative Couse Schedule – Syllabus)

TOBB ETÜ
Ekonomi ve Teknoloji Üniversitesi

1. Hafta: Sayısal Sinyaller/Sistemler, İkilik Tabanda Sayılar, Taban Aritmetiği, İşaretli/Eksi Sayıların Gösterimi, Sayısal Tasarım Tarihçesi

2. Hafta: İkili Mantık Aritmetiği ve Kapıları, Bool Cebiri Teorisi ve Tanımları, Bool Fonksiyonları, Kapı-Seviyesinde Yalınlaştırma, Karnough Haritası, Önemsenmeyen Durumlar, NAND, NOR, XOR

3-4. Hafta: FPGA, Birleşik (Combinational) Devreler, Aritmetik Modüller, Decoder, Encoder, Mux, Verilog HDL

5. Hafta: Ardışık (Sequential) Devreler, Mandal (Latch), Flip-Flop, Yazmaçlar (Registers)

Lab Sınavı (265/264L)

6. Hafta: Durum Makinaları, Örnek Tasarımlar, Sayaçlar (Counters)

7. Hafta: FSM Örnekleri

8. Hafta: **(31 Ekim – 4 Kasım)** RTL (Register Transfer Level) ASMD (Algorithmic State Machine and Datapath) Tasarımları

9. Hafta: **(7-11 Kasım)** Durağan Zaman Analizi (Static Timing Analysis)

Ara Sınav (265/264) **(15 Kasım)**

10. Hafta: **(14-18 Kasım)** Bellekler, FPGA'da Block RAM, OpenRAM

11-12. Hafta: **(21-25 Kasım, 28 Kasım – 2 Aralık)** Boru hattı, FPGA ve ASIC Tasarım Akışları

Final **(Aralık)** – Proje Teslimleri **(18 Aralık)**

REGISTER TRANSFER LEVEL (RTL)

The information flow and processing performed on the data stored in the registers are referred to as register transfer operations

A digital system is represented at the register transfer level (RTL) when it is specified by the following three components:

1. The set of registers in the system.
2. The operations that are performed on the data stored in the registers.
3. The control that supervises the sequence of operations in the system.

REGISTER TRANSFER LEVEL (RTL) ÖRNEKLERİ

TOBB ETÜ
Ekonomi ve Teknoloji Üniversitesi

R2 ← R1 ; R1 yazmaç içeriğini R2 yazmacına yaz

if (T1 = 1) then (R2 ← R1) ; Eğer T1 1'e eşitse R1 yazmacını R2'ye yaz

if (T3 = 1) then (R2 ← R1, R1 ← R2) ; Eğer T3 1'e eşitse, R1 ve R2 yazmaçlarının değerlerini değiştir

R1 ← R1 + R2 ; R1 ve R2 yazmaç değerlerini topla R2'ye yaz

R3 ← R3 + 1 ; R3 yazmaç değerini 1 arttır

R4 ← shr R4 ; R4 yazmacını sağa bir bit kaydır

R5 ← 3 ; R5 yazmacına '3' değerini yaz

REGISTER TRANSFER LEVEL (RTL) ÖRNEKLERİ

TOBB ETÜ
Ekonomi ve Teknoloji Üniversitesi

Sayısal sistemlerde genelde karşılaşılan 4 farklı operasyon:

1. Transfer operasyonları: Bir yazmacın verisini başka yazmaca kopyalamak
2. Aritmetik operasyonlar: Toplama, çıkarma, çarpma, bölme
3. Mantıksal operasyonlar: AND, OR, XOR, NAND, NOR
4. Kaydırma (shift) operasyonları

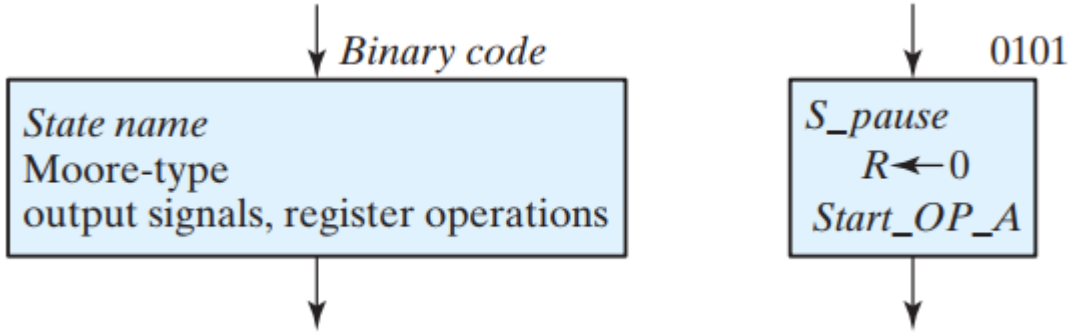
Algoritmik Durum Makinası (Algorithmic state machine)

Bir sayısal sistemdeki veriler veri (data) ve kontrol (control) olarak sınıflandırılabilir

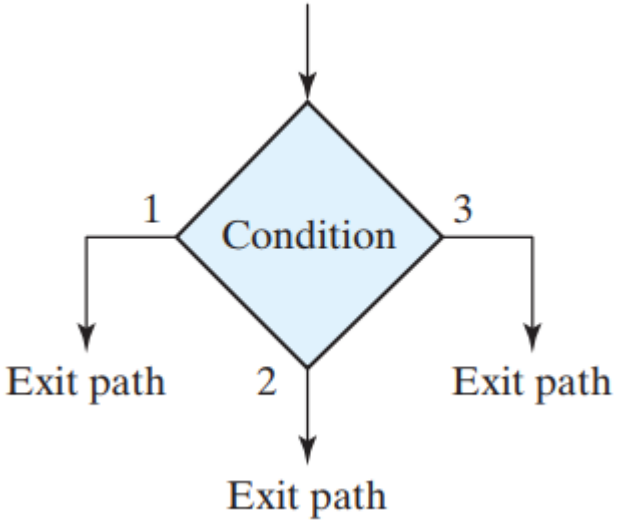
Veriler üzerinde çeşitli aritmetik, mantıksal, kaydırma veya diğer benzer işlemler yapılır

Kontrol sinyalleri, veriler üzerinde ne tür işlemlerin nasıl bir sıralama ile ve hangi durumlarda yapılacaklarını kontrol ederler

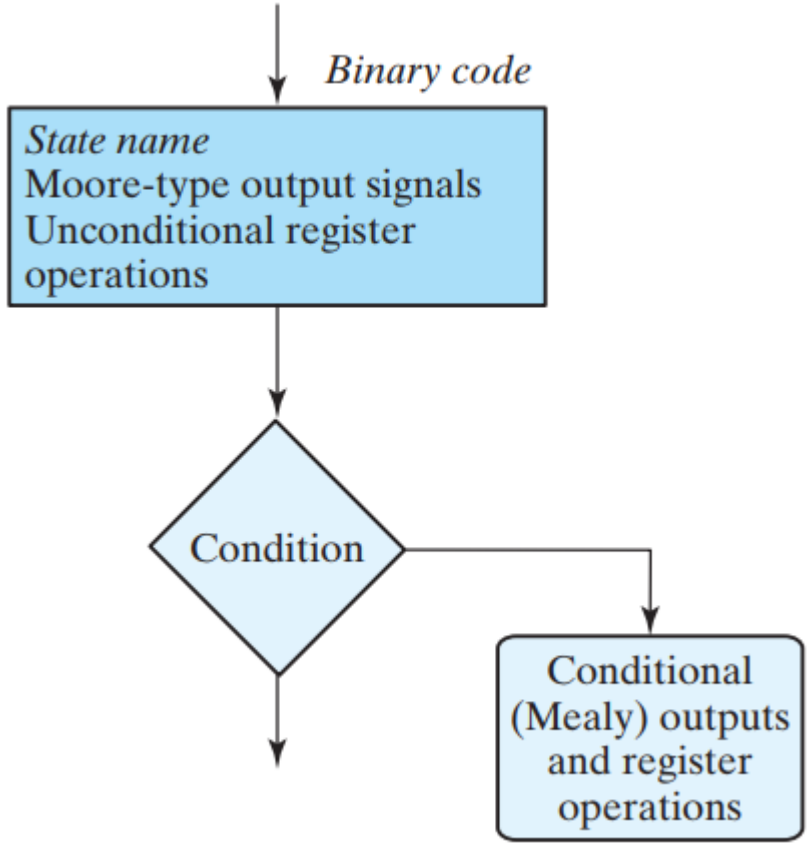
ALGORİTMİK DURUM MAKİNASI



Durum Kutusu (State Box)



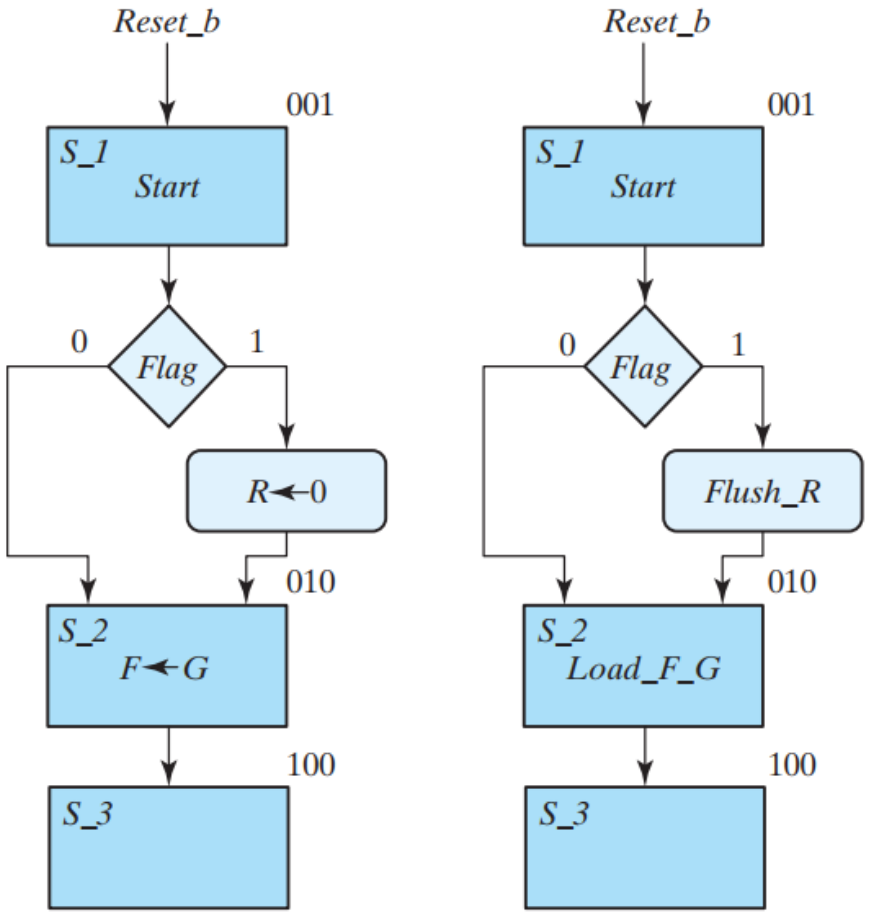
Karar Kutusu (Decision Box)

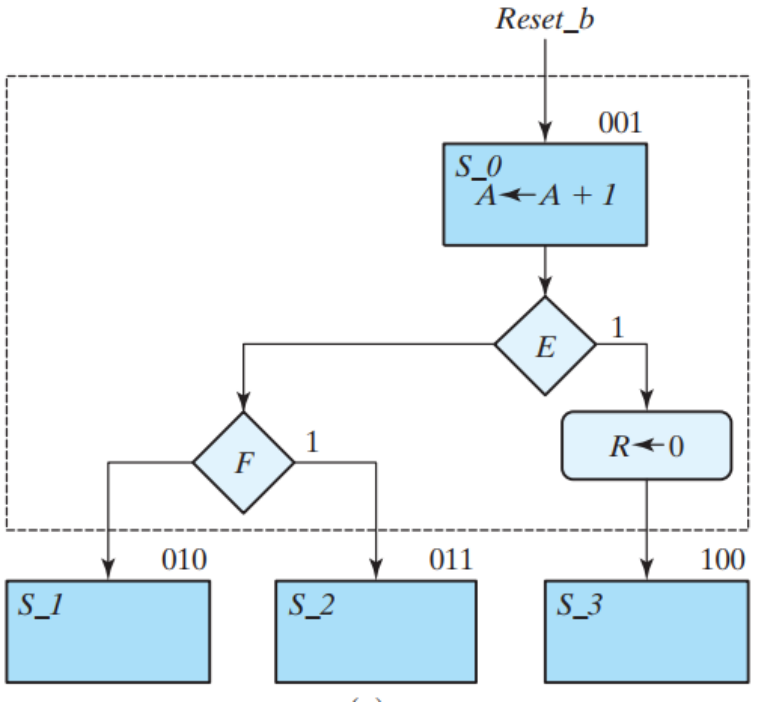


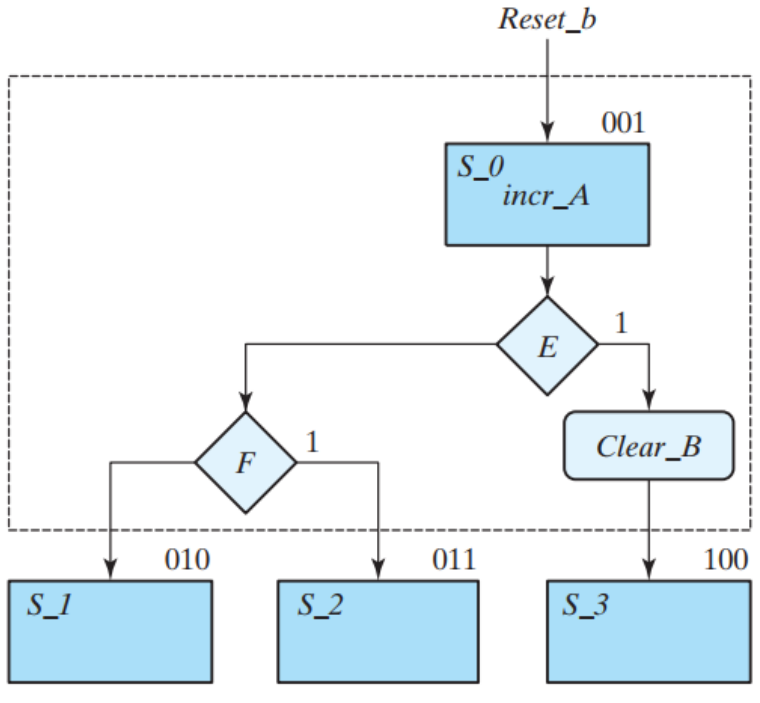
Koşullu Kutu (Conditional Box)

ALGORİTMİK DURUM MAKİNASI ÖRNEKLERİ

TOBB ETÜ
Ekonomi ve Teknoloji Üniversitesi







ALGORİTMİK DURUM MAKİNASI ÖRNEKLERİ

TOBB ETÜ
Ekonomi ve Teknoloji Üniversitesi

Basit bir otopilot kontrol algoritması:

Bir İHA aracı GNSS alıcı ve altimetre'den aldığı verilere göre otomatik olarak yerden yükselecek ve belirtilen mesafede sabit duracaktır

Yükselinecek olan mesafe, sisteme güç verildikten sonra seri arayüzden yüklenecektir

Yükleme talebi yapılan mesafe 10 metre ile 100 metre arasında olmalıdır, aksi takdirde seri arayüzden cevap olarak hata kodu dönülecek ve yükleme talebinde kalınacaktır

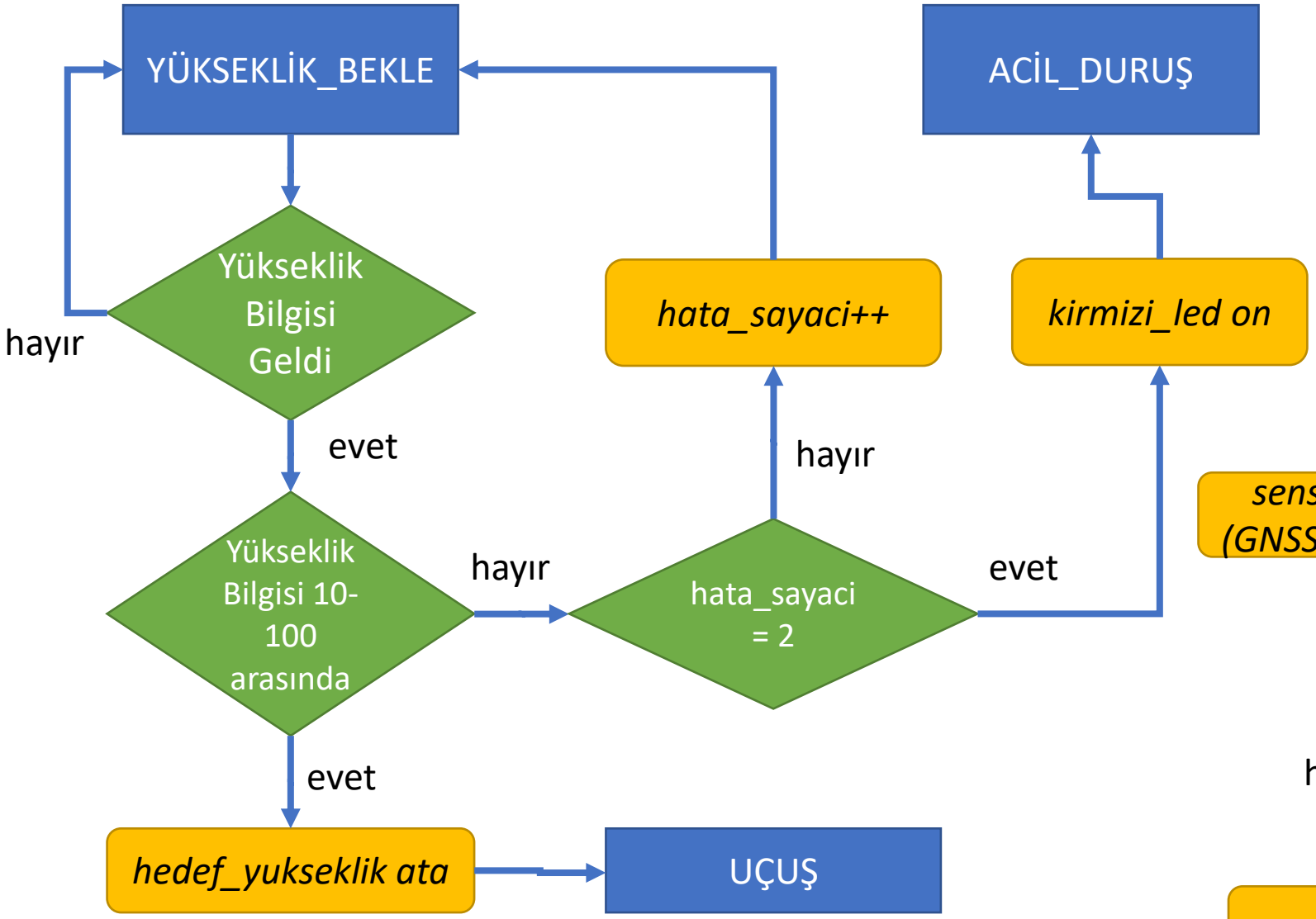
3 kez üst üste yanlış yükleme talebi gelirse system acil duruş durumuna geçip kırmızı ledi yakacaktır

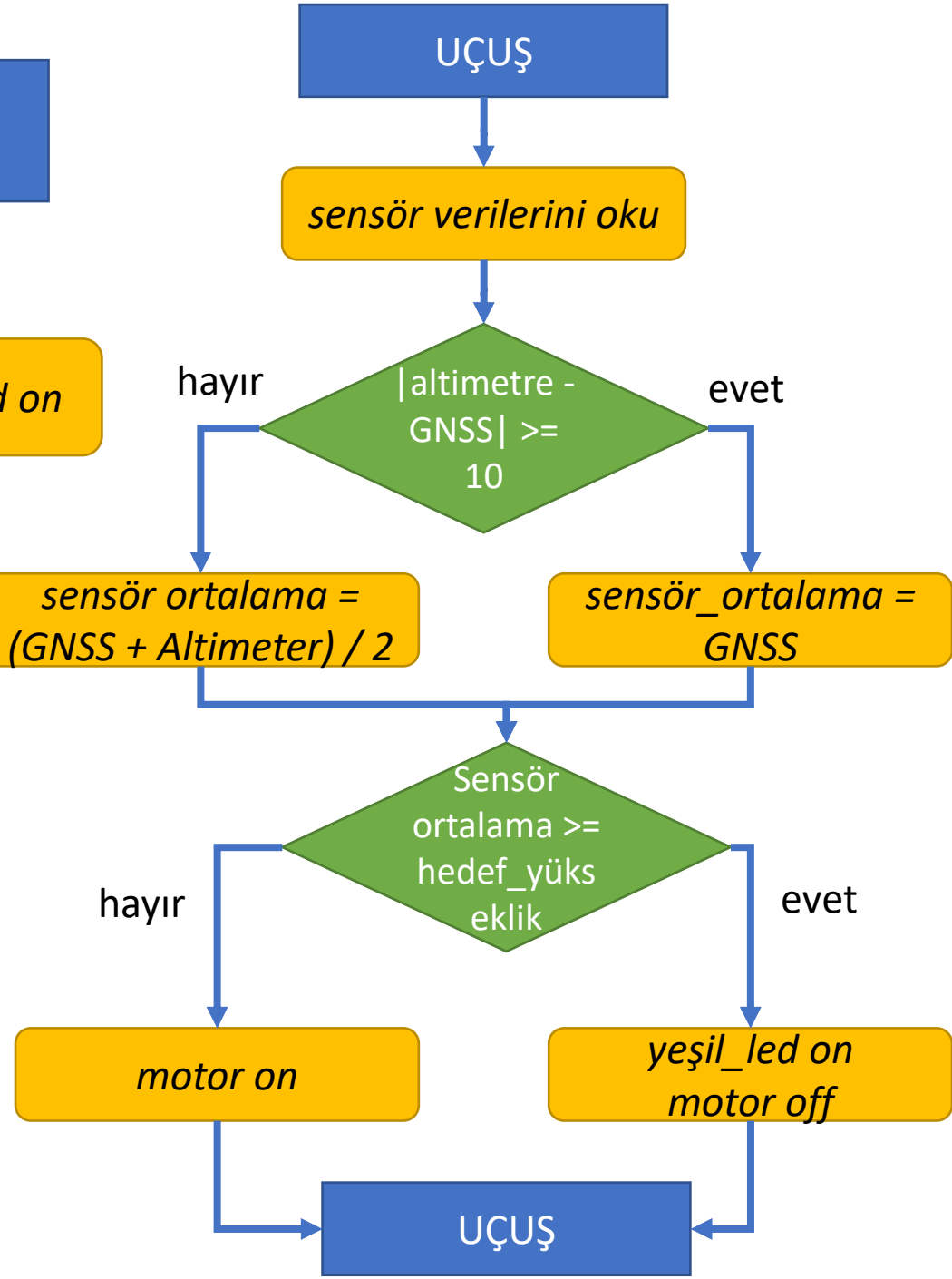
Yükseklik ölçümü için iki adet sensör verilerinin ortalamasını kullanacaktır

İki adet sensörün verileri arasındaki fark 10 metre veya fazla ise GNSS verisini kabul edip altimetre verisini hesaba katmayacaktır

Yükseklik verisi belirlenen hedeften az ise motora güç vermeye devam edecektir, hedefe ulaşıldıysa ya da hedef aşıldıysa motor kapatılacaktır

Hedefe ilk ulaşıldığında yeşil led yakılacaktır





TOBB ETÜ
Ekonomi ve Teknoloji Üniversitesi

```
module fsm_otopilot
(
input clk,
input rst_n,
input [15:0] gnss_i,
input [15:0] altimetre_i,
input [7:0] hedef_yukseklik_i,
input yukseklik_bilgisi_i,
output motor_o,
output yesil_led_o,
output kirmizi_led_o
);

localparam S_YUKSEKLIK_BEKLE    = 2'b00;
localparam S_ACIL_DURUS         = 2'b01;
localparam S_UCUS               = 2'b10;

// combinational assignments
reg hedef_yukseklik_hata;
reg motor_ac;
reg [15:0] sensor_fark;
reg [15:0] sensor_ortalama;

// registers
reg [1:0] state;
reg [1:0] hata_sayaci;
reg [7:0] atanan_yukseklik;
reg yesil_led;
reg kirmizi_led;
reg motor;
```

```
always @(*) begin
    if (hedef_yukseklik_i < 10 || hedef_yukseklik_i > 100) begin
        hedef_yukseklik_hata = 1'b1;
    end
    else begin
        hedef_yukseklik_hata = 1'b0;
    end

    if (gnss_i > altimetre_i) begin
        sensor_fark = gnss_i - altimetre_i;
    end
    else begin
        sensor_fark = altimetre_i - gnss_i;
    end

    if (sensor_fark > 9) begin
        sensor_ortalama     = gnss_i;
    end
    else begin
        sensor_ortalama     = (gnss_i + altimetre_i) >> 1;
    end

    motor_ac = 0;
    if (state == S_UCUS) begin
        if (sensor_ortalama >= atanan_yukseklik) begin
            motor_ac = 1'b0;
        end
        else begin
            motor_ac = 1'b1;
        end
    end

end
```

```
always @(posedge clk, negedge rst_n) begin
    if (rst_n == 1'b0) begin
        state               <= S_YUKSEKLIK_BEKLE;
        hata_sayaci         <= 2'b00;
        atanan_yukseklik    <= 8'b00000000;
        yesil_led           <= 1'b0;
        kirmizi_led         <= 1'b0;
        motor               <= 1'b0;
    end
    else begin
        case (state)
            S_YUKSEKLIK_BEKLE : begin
                if (yukseklik_bilgisi_i == 1'b1) begin
                    if (hedef_yukseklik_hata == 1'b0) begin
                        atanan_yukseklik    <= hedef_yukseklik_i;
                        state               <= S_UCUS;
                    end
                    else begin
                        if (hata_sayaci == 2'b10) begin
                            state       <= S_ACIL_DURUS;
                            kirmizi_led <= 1'b1;
                        end
                        else begin
                            hata_sayaci <= hata_sayaci + 2'b01;
                            state       <= S_YUKSEKLIK_BEKLE;
                        end
                    end
                end
                else begin
                    state <= S_YUKSEKLIK_BEKLE;
                end
            end
            S_ACIL_DURUS : begin
                //
            end
            S_UCUS : begin
                if (motor_ac == 1'b1) begin
                    motor       <= 1'b1;
                end
                else begin
                    motor       <= 1'b0;
                    yesil_led   <= 1'b1;
                end
            end

            default: begin
                state <= S_YUKSEKLIK_BEKLE;
            end
        endcase
    end
end

assign motor_o          = motor;
assign yesil_led_o      = yesil_led;
assign kirmizi_led_o    = kirmizi_led;
```

Resource	Estimation	Available	Utilization %
LUT	74	20800	0.36
FF	15	41600	0.04
IO	46	106	43.40
BUFG	1	32	3.13

Sources | Netlist

- FSM_onehot_state[2]_i_3 (LUT6)
- FSM_onehot_state[2]_i_4 (LUT6)
- FSM_onehot_state[2]_i_5 (LUT6)
- FSM_onehot_state_reg[0] (FDPE)
- FSM_onehot_state_reg[1] (FDCE)
- FSM_onehot_state_reg[2] (FDCE)
- GND (GND)

```
localparam S_YUKSEKLIK_BEKLE     = 2'b00;
localparam S_ACIL_DURUS          = 2'b01;
localparam S_UCUS                = 2'b10;
```

```
// registers
reg [1:0] state;
```

Options

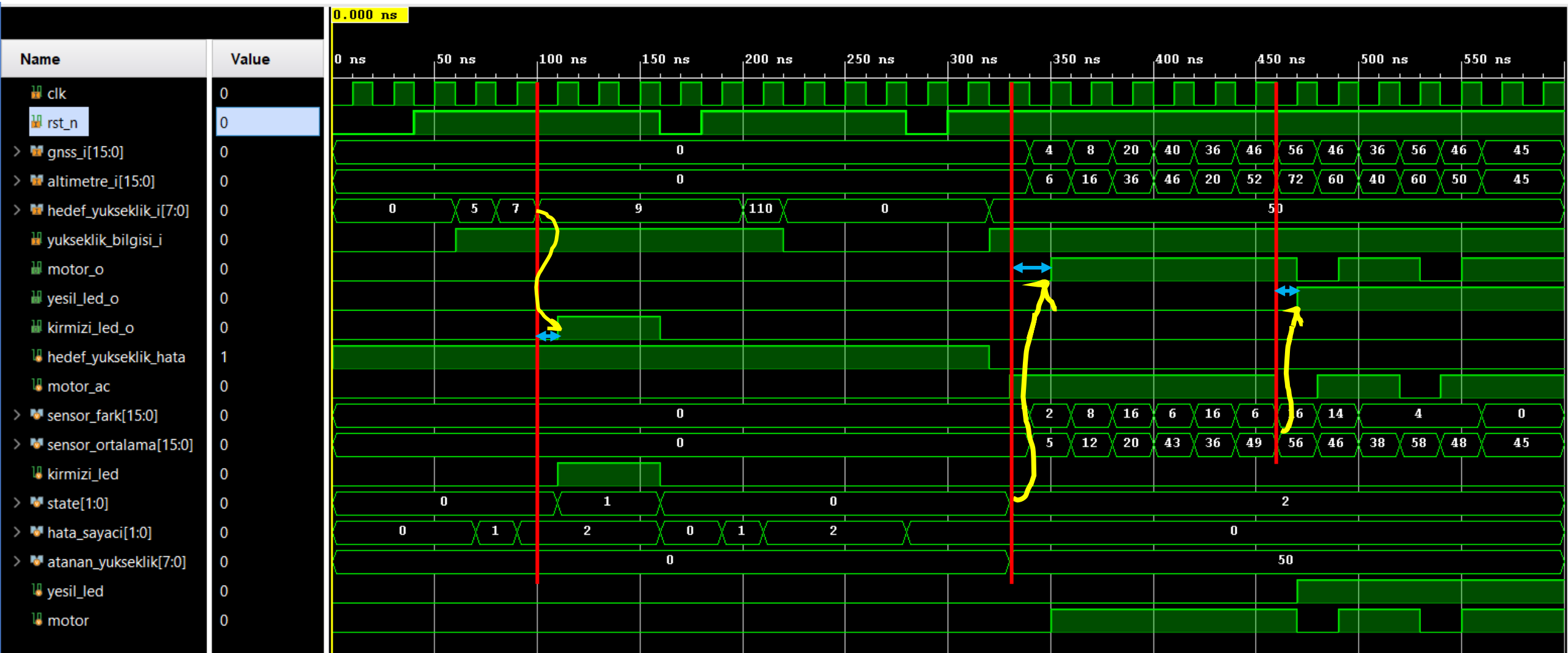
-flatten_hierarchy	rebuilt
-gated_clock_conversion	off
-bufg	12
-fanout_limit	10,000
-directive	Default
-retiming	
-fsm_extraction	auto
-keep_equivalent_registers	
-resource_sharing	
-control_set_opt_threshold	
-no_lc	
-no_srlextract	
-shreg_min_size	3

auto
off
one_hot
sequential
johnson
gray
auto

Resource	Estimation	Available	Utilization %
LUT	73	20800	0.35
FF	14	41600	0.03
IO	46	106	43.40
BUFG	1	32	3.13

- FSM_sequential_state_reg[0] (FDCE)
- FSM_sequential_state_reg[1] (FDCE)

Resource	Estimation	Available	Utilization %
LUT	74	20800	0.36
FF	15	41600	0.04
IO	46	106	43.40
BUFG	1	32	3.13



ALGORİTMİK DURUM MAKİNASI ÖRNEKLERİ

TOBB ETÜ
Ekonomi ve Teknoloji Üniversitesi

ADAS (Advanced Driver Assistance System) ve Otonom Sürüş Örneği:
Yeni piyasaya çıkarılacak olan bir araçta hem assistance hem de otonom sürüş modları bulunmaktadır.

Assistance durumunda sürüşe doğrudan müdahale edilmeyecek, sadece ışıklı uyarılar yapılacak. Mod geçişleri bir sinyal aracılığıyla gerçekleşecektir.

Otonom sürüş durumunda ise sürücü müdahale edene kadar araç kendi kendine sensörlerden aldığı bilgilere göre fren ve gazı aktive ederek sürüşü gerçekleştirecektir. Otonom'dan assistance durumuna geçerken gaz ve fren inaktive edilir.

Araç ilk çalıştırıldığında assistance durumunda başlayacaktır. Otonom moda geçmeden önce sürücü geçiş durumuna alıp, takip mesafesi ve ortalama hız bilgilerini girecektir.

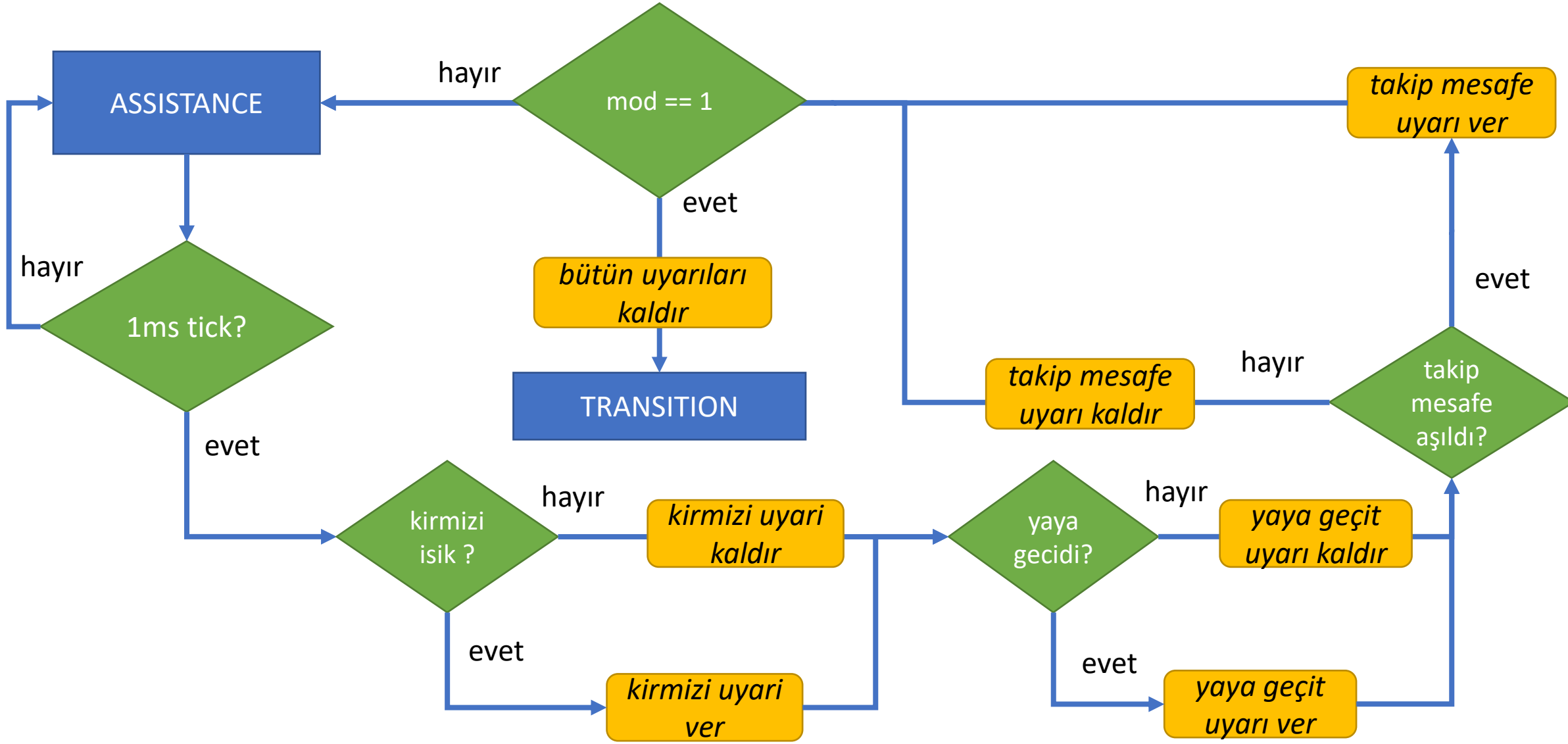
Otonom sürüş sırasında lidar ve kamera sensörlerinden kırmızı ışık, yaya geçidi, öndeki araçla mesafe bilgilerini her 1 ms'de bir alacaktır. Kırmızı ışık ve yaya geçidi bilgilerinde iki sensörün de pozitif olması gerekmektedir, mesafede ise iki sensörün ortalaması alınacaktır. Eğer iki sensörün ölçümleri arasındaki fark 20 mt'den fazla ise lidar mesafesi geçerli sayılacaktır. Takip mesafe default olarak 50 mt ve hız 100 km/s olarak başlatılacaktır.

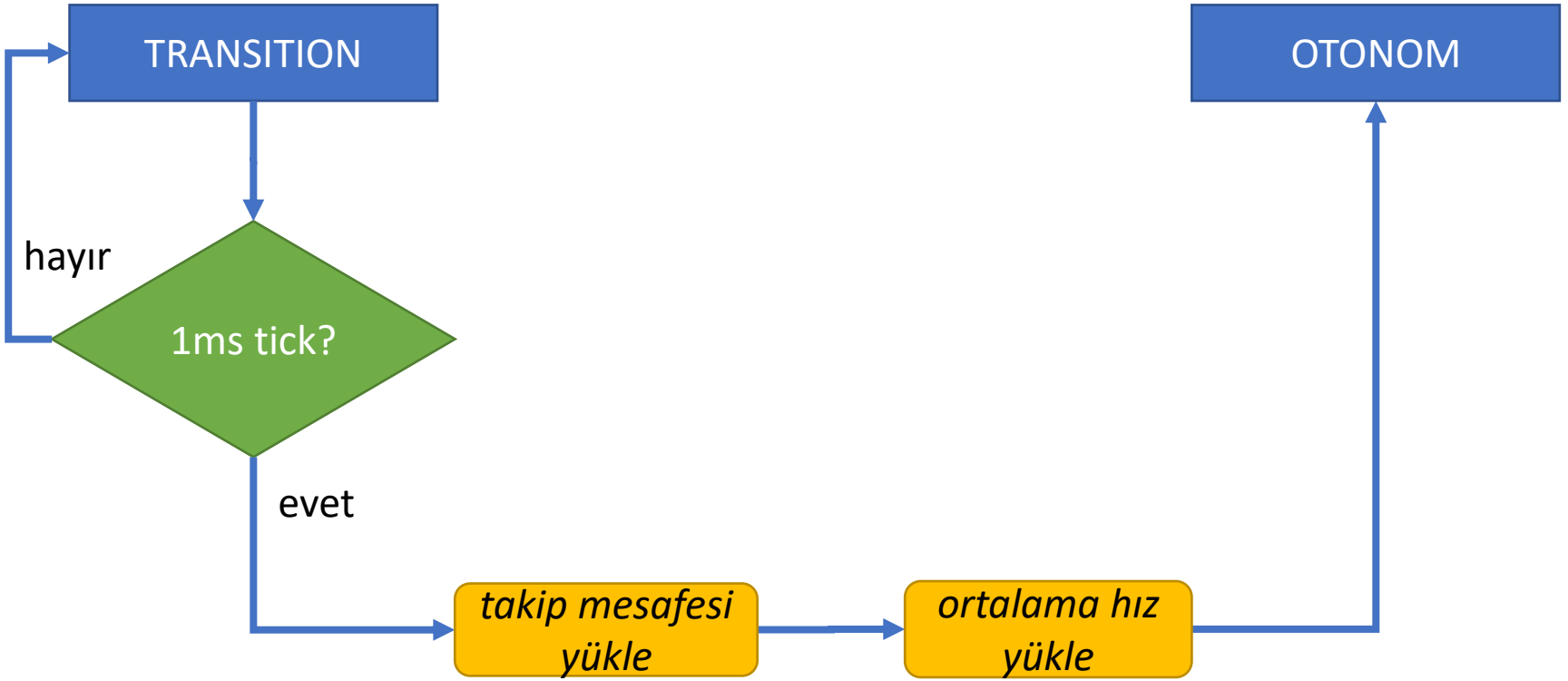
Kırmızı ışık tespit edildiğinde araç duracak, yaya geçidi tespit edildiğinde araç hızı 20 km/s'e düşürülecektir. Takip mesafesi ölçümlerinde son 4 ölçümün ortalaması alınacak ve buna göre gaz ya da fren aktive edilecektir. Eğer bir engel yoksa, araç istenilen hıza ulaşana kadar gaza basılacaktır. Gaz ve fren ikisine de basılmadığında hızın sabit kaldığı varsayılacaktır.

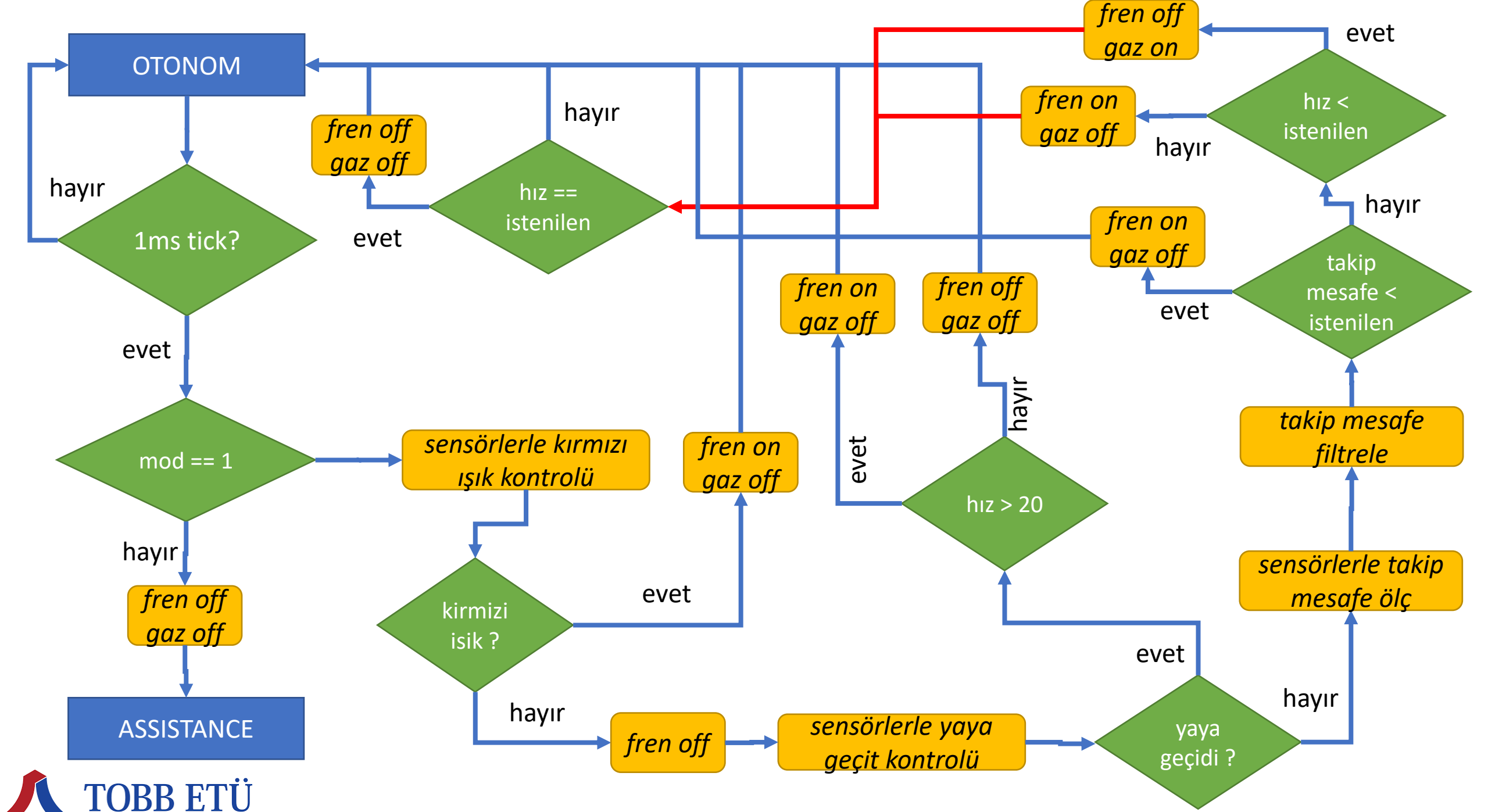
Aracın hızı ölçülebiliyor olacaktır. Sistemin bütün girdi ve çıktıları 1ms'lik timer tick ile alınacak ve aktive olacaktır.

```
module fsm_adas
(
input clk,
input rst_n,
input timer_tick_i,                 // her 1 ms'de 1 kez '1' olur
input mod_i,                        // 1'b0 -> assistance 1'b1 otonom
input [1:0] kirmizi_isik_i,         // (1) lidar, (0) kamera
input [1:0] yaya_gecidi_i,          // (1) lidar, (0) kamera
input [7:0] mesafe_olcum_lidar_i,   // lidar ondeki arac olcumu (metre)
input [7:0] mesafe_olcum_kamera_i,  // kamera ondeki arac olcumu (metre)
input [7:0] mesafe_giris_i,         // otonom mod icin arac takip mesafe girdisi
input [7:0] hiz_olcum_i,            // aracin hızı olcum (km/s)
input [7:0] hiz_giris_i,            // otonom mod icin arac hiz girdisi
output gaz_o,                       // otonom mod icin gaza bas
output fren_o,                      // otonom mod icin frene bas
output kirmizi_isik_o,              // assistance mod icin uyari
output yaya_gecidi_o,               // assistance mod icin uyari
output takip_mesafe_o               // assistance mod icin uyari
);

endmodule
```


ASSISTANCE
hayır
mod == 1
takip mesafe uyarı ver
evet
bütün uyarıları kaldır
hayır
1ms tick?
TRANSITION
evet
takip mesafe uyarı kaldır
hayır
takip mesafe aşıldı?
evet
kirmizi isik ?
hayır
kirmizi uyari kaldır
yaya gecidi?
hayır
yaya geçit uyarı kaldır
evet
kirmizi uyari ver
evet
yaya geçit uyarı ver


TRANSITION
OTONOM
hayır
1ms tick?
evet
takip mesafesi yükle
ortalama hız yükle



TOBB ETÜ
Ekonomi ve Teknoloji Üniversitesi

```
// Gray encoding for FSM regs
localparam ASSISTANCE   = 2'b00;
localparam TRANSITION   = 2'b01;
localparam OTONOM       = 2'b11;

localparam ON   = 1'b1;
localparam OFF  = 1'b0;

integer i;

// registers
reg [7:0] olcum_buffer [0:2];
reg [1:0] state;
reg [7:0] takip_mesafe_kaydedilen;
reg [7:0] hiz_kaydedilen;

// comb signals
reg kirmizi_isik_var;
reg yaya_gecidi_var;
reg [7:0] mesafe_fark;
reg [8:0] yeni_mesafe_olcum;    // saga kaydirma nedeniyle 1 bit fazla
reg [9:0] olcum_ortalama;       // saga kaydirma nedeniyle 2 bit fazla

// for output signals
reg gaz, fren, kirmizi_isik, yaya_gecidi, takip_mesafe;
```

```
always @(*) begin

    kirmizi_isik_var = 1'b0;
    if (kirmizi_isik_i == 2'b11) begin
        kirmizi_isik_var = 1'b1;
    end

    yaya_gecidi_var = 1'b0;
    if (yaya_gecidi_i == 2'b11) begin
        yaya_gecidi_var = 1'b1;
    end

    if (mesafe_olcum_lidar_i >= mesafe_olcum_kamera_i) begin
        mesafe_fark = mesafe_olcum_lidar_i - mesafe_olcum_kamera_i;
    end
    else begin
        mesafe_fark = mesafe_olcum_kamera_i - mesafe_olcum_lidar_i;
    end

    if (mesafe_fark < 20) begin
        yeni_mesafe_olcum = (mesafe_olcum_lidar_i + mesafe_olcum_kamera_i) >> 1;
    end
    else begin
        yeni_mesafe_olcum = mesafe_olcum_lidar_i;
    end

    olcum_ortalama = (olcum_buffer[2] + olcum_buffer[1] + olcum_buffer[0] + yeni_mesafe_olcum) >> 2;

end
```

```
always @(posedge clk, negedge rst_n) begin
    if (rst_n == 1'b0) begin
        state                       <= ASSISTANCE;
        gaz                         <= 1'b0;
        fren                        <= 1'b0;
        kirmizi_isik                <= 1'b0;
        yaya_gecidi                 <= 1'b0;
        takip_mesafe                <= 1'b0;
        takip_mesafe_kaydedilen     <= 50;
        hiz_kaydedilen              <= 100;
        for (i = 0; i < 3 ; i=i+1) begin
            olcum_buffer[i]         <= {8{1'b0}};
        end
    end
```

```
else begin
    case (state)

        ASSISTANCE : begin
            if (timer_tick_i == 1'b1) begin

                if (kirmizi_isik_var == 1'b1) begin
                    kirmizi_isik    <= ON;
                end
                else begin
                    kirmizi_isik    <= OFF;
                end

                if (yaya_gecidi_var == 1'b1) begin
                    yaya_gecidi     <= ON;
                end
                else begin
                    yaya_gecidi     <= OFF;
                end

                if (olcum_ortalama < takip_mesafe_kaydedilen) begin
                    takip_mesafe    <= OFF;
                end
                else begin
                    takip_mesafe    <= ON;
                end

                if (mod_i == 1'b1) begin
                    kirmizi_isik    <= OFF;
                    yaya_gecidi     <= OFF;
                    takip_mesafe    <= OFF;
                    state           <= TRANSITION;
                end

            end
        end
```

```
TRANSITION : begin
    if (timer_tick_i == 1'b1) begin
        takip_mesafe_kaydedilen <= mesafe_giris_i;
        hiz_kaydedilen          <= hiz_giris_i;
        state                   <= OTONOM;
    end
end
```

```
OTONOM : begin
    if (timer_tick_i == 1'b1) begin

        if (mod_i == 1'b0) begin
            fren    <= OFF;
            gaz     <= OFF;
            state   <= ASSISTANCE;
        end
        else begin

            olcum_buffer[2] <= olcum_buffer[1];
            olcum_buffer[1] <= olcum_buffer[0];
            olcum_buffer[0] <= olcum_ortalama;

            if (olcum_ortalama < takip_mesafe_kaydedilen) begin
                fren    <= ON;
                gaz     <= OFF;
            end
            else begin
                if (hiz_olcum_i < hiz_kaydedilen) begin
                    fren    <= OFF;
                    gaz     <= ON;
                end
                else if (hiz_olcum_i == hiz_kaydedilen) begin
                    fren    <= OFF;
                    gaz     <= OFF;
                end
                else begin
                    fren    <= ON;
                    gaz     <= OFF;
                end
            end
```

```
if (yaya_gecidi_var == 1'b1) begin
    if (hiz_olcum_i > 20) begin
        fren    <= ON;
        gaz     <= OFF;
    end
    else begin
        fren    <= OFF;
        gaz     <= OFF;
    end
end

if (kirmizi_isik_var == 1'b1) begin
    fren    <= ON;
    gaz     <= OFF;
end
```

```
default: begin
    state <= ASSISTANCE;
end
```

```
// output signal assignments
assign gaz_o            = gaz;
assign fren_o           = fren;
assign kirmizi_isik_o   = kirmizi_isik;
assign yaya_gecidi_o    = yaya_gecidi;
assign takip_mesafe_o   = takip_mesafe;
```

Resource	Estimation	Available	Utilization %
LUT	67	20800	0.32
FF	47	41600	0.11
IO	53	106	50.00
BUFG	1	32	3.13