T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

FPGA VE MİKRODENETLEYİCİNİN PC İLE SERİ HABERLEŞMESİ PERFORMANS KARŞILAŞTIRMASI

YAHYA TAŞTAN
YÜKSEK LİSANS TEZİ
ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

GEBZE
2019

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

FPGA VE MİKRODENETLEYİCİNİN PC İLE SERİ HABERLEŞMESİ PERFORMANS KARŞILAŞTIRMASI

YAHYA TAŞTAN
YÜKSEK LİSANS TEZİ
ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMANI
DR. ÖĞR. ÜYESİ ÖNDER ŞUVAK

GEBZE
2019

T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

PERFORMANCE COMPARISON WITH PC FPGA AND MICROCONTROLLER SERIAL COMMUNICATION

YAHYA TAŞTAN

A THESIS SUBMITTED FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRONIC ENGINEERING

THESIS SUPERVISOR

ASSIST. PROF. DR. ONDER SUVAK

GEBZE

2019

GEBZE TEKNİK ÜNİVERSİTESİ	YÜKSEK LİSANS JÜRİ ONAY FORMU

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 16/1/2019 tarih ve 2019/04 sayılı kararıyla oluşturulan jüri tarafından 18/1/2019 tarihinde tez savunma sınavı yapılan Yahya TAŞTAN'ın tez çalışması Elektronik Mühendisliği Anabilim Dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir.

JÜRİ

ÜYE (TEZ DANIŞMANI)	: Dr. Önder ŞUVAK
ÜYE	: Doç. Dr. Serdar Süer ERDEM
ÜYE	: Dr. Mustafa Berke YELTEN

ONAY

Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun

...../...../.....tarih ve ...../..... sayılı kararı.

ÖZET

Evrensel asenkron alıcı vericisi (UART), bilgisayarlar ve diğer çevresel aygıtlar arasında seri iletişim için kullanılan bir cihazdır. Bu proje bir evrensel asenkron alıcı verici (UART) tasarımını açıklar. Bu çalışma üç parçadan oluşur: Birinci parçada; VERILOG dili kullanılarak FPGA üzerinde çalışacak bir UART protokolü programlandı. İkinci parçada bir mikrodenetleyici, C dili kullanılarak hazır kütüphanelerden UART tasarlandı. Üçüncü parça ise mikrodenetleyici ve FPGA arasında haberleşme sağlayacak olan bilgisayardaki C# dilinde yazılan arayüz panelidir. Bu yüksek lisans tezi iki kısımdan oluşur. Birinci kısımda bilgisayardaki seri iletişim arayüz programından iki basamaklı iki sayı FPGA'e gönderildi. Bu sayılar seven segment displayde gösterildi. Bu iki sayının toplamları seven segment displayde gösterilip tekrar bilgisayardaki seri iletişim arayüz programına gönderildi. Seri iletişim programı sonucu mikrodenetleyiciye gönderip seven segment displayde gösterildi. Sistem mikrodenetleyici-bilgisayar-FPGA sırasıyla da çalıştırılabilir. Bilgisayardaki seri iletişim arayüz programı ile sistem sıfırlanabilir. İkinci kısımda ise mikrodenetleyicinin ölçtüğü sıcaklık değerleri bilgisayardaki seri iletişim arayüz programı aracılığıyla FPGA gönderildi. FPGA bu sıcaklık değerlerinin ortalamasını alarak bilgisayardaki seri iletişim arayüz programına gönderdi. FPGA'in yapması gereken işler mikrodenetleyici tarafından yapılarak bilgisayar arayüz programı aracılığıyla FPGA'e hazır olarak verilmiştir. Bu şekilde FPGA'in iş yükü ve maaliyeti azaltıldığı gösterilmiştir.

Anahtar Kelimeler: Evrensel Asenkron Alıcı Verici (UART), VERILOG, Alanda Programlanabilir Kapı Dizisi (FPGA), mikrodenetleyici.

SUMMARY

The Universal Asynchronous Receiver Transmitter (UART) is a device used for serial communication between computers and other peripheral devices. This project describes the design of a universal asynchronous transceiver (UART). This study consists of three parts: In the first part; A UART protocol was programmed to work on the FPGA using the VERILOG language. In the second part, the UART was designed using a microcontroller C language. The third part is the interface panel written in C# on the computer that will provide communication between the microcontroller and FPGA. This master's thesis consists of two parts. In the first part, two numbers were sent to FPGA from the serial communication interface program on the computer. These numbers are shown on the segment display. The totals of these two numbers were displayed in the segment display and sent back to the serial communication interface program on the computer. The serial communication program was sent to the microcontroller and was shown on the segment display. The system microcontroller-computer-FPGA can also be operated respectively. The system can be reset with the serial communication interface program on the computer. In the second part, the temperature values measured by the microcontroller are sent to the FPGA via the serial communication interface program on the computer. The FPGA received the average of these temperature values and sent it to the serial communication interface program on the computer. The tasks that FPGA should do are made by microcontroller and ready to FPGA by computer interface program. In this way, the workload and cost of FPGA has been shown to be reduced.

Key Words: Universal Asynchronous Receiver-Transmitter (UART), VERILOG, Field Programmable Gate Array (FPGA), mikrocontroller.

TEŞEKKÜR

Başta, yüksek lisans eğitimimde ve bu tez çalışmamda bana yol gösteren, desteğini ve yardımlarını hiçbir zaman esirgemeyip bilgisi ile bu çalışmanın oluşmasının yolunu açan danışmanım Dr. Önder ŞUVAK'a,

Ve göstermiş oldukları maddi ve manevi desteklerinden dolayı aileme en içten teşekkürlerimi sunarım.

İÇİNDEKİLER

SİMGELER ve KISALTMALAR DİZİNİ

Simgeler ve Kısaltmalar		Açıklamalar
sn	:	Saniye
Hz	:	Hertz
MHz	:	Mega Hertz
ASIC	:	Application-Specific Integrated Circuit
ASCII	:	American Standard Code for Information Interchange
CCS	:	Custom Computer Services
FIFO	:	First In, First Out
FSM	:	Finite State Machine
FPGA	:	Field Programmable Gate Array
GTÜ	:	Gebze Teknik Üniversitesi
uC	:	µ-controller
MİB	:	Merkezi İşlem Birimi
MCU	:	Microcontroller Unit
JTAG	:	Joint Test Action Group
I/O	:	Input/Output
SPI	:	Serial Peripheral Interface
OTPROM	:	One-Time Programmable Read-Only Memory
LCD	:	Liquid-Crystal Display
RTL	:	Register-Transfer Level
UART	:	Universal Asynchronous Receiver/Transmitter
VGA	:	Video Graphics Array
VHDL	:	Very High Speed Integrated Circuit Hardware Description Language

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

1. GİRİŞ

Mantık IC için hızlı dönüş prototip sistem geliştirme ve kritik pazar taleplerini karşılamaktır. Günümüzde bir tasarımcı ASIC veya FPGA teknolojisinde mantığı uygular. FPGA tasarım ve esneklik için iyi bir yoldur. Bir ürünü ve dolayısıyla sistemi hızlı bir şekilde geliştirmek için prototip ve esnek IC çözümü için FPGA hızlı bir şekilde iyi bir çözüm sunar. FPGA düşük hacimli üretim için düşük maliyet avantajı sunar ve yeniden programlanabilir [1].

Prototip oluşturma, gelecekteki yükseltme ve değiştirme için VERILOG'da kodlama hem basit hem de kolay anlaşılırdır. Mantık devresi veya kod kolayca doğrulanabilir, düzeltmeler hızlıdır ve sonuç anında simüle edilebilir. FPGA endüstride dijital devre tasarım uygulamasında bir standart sağlar. Ayrıca değişik kod tasarımını test etmek için basit bir yol sağlar ve kodu analiz edebilir ve optimizasyon yapabilir.

Bitlerin bir yerden başka bir yere taşınması için kablo veya başka bir ortam kullanılırken her fazla metre kablo giderleri arttırması seri iletişimde bir sorundur. İletişim bağlantılarının giderlerini azaltmak için paralel olarak taşınması gereken veri bitleri seri olarak gönderilir. Bağlantının diğer ucunda da iletilen bitleri seriden paralel hale dönüştürmek için UART kullanılmaktadır. Evrensel asenkron alıcı verici olarak adlandırılan UART, paralel ve seri bitler arasında çeviri yapan bir bilgisayar donanımı parçasıdır. UART, I/O veriyolu ve seri aygıt arasındaki arabirim gibi davranır. Öneğin; bir bilgisayarın arabirimini fare veya modem gibi seri aygıtını kontrol eder. Bu projede VERILOG donanım programla dili kullanılarak FPGA üzerinde UART tasarımı uygulandı.

Garima Bandhawarkar Wakhle ve arkadaşları FPGA üzerinde gerçeklenen UART modülü tasarımını gerçekleştirmiştir. İlgili çalışmada UART modülü receiver ve transmitter olmak üzere iki kısımda tasarımı yapılmış ve frekans bölücü devre de tasarıma eklenmiştir. Böylece istenen baud rate oranında veri aktarımı mümkün kılınmıştır. Yine bu çalışmada, RS232 seri haberleşme tasarımı için ilgili devrenin her iki modülünün durum diyagramları ve simulasyon sonuçları gerçeklenmiştir [8].

Nurul Fatihah Jusoh ve arkadaşları FPGA tabanlı yaptıkları çalışmada RS232-USB çevirici üzerinde çalışmıştır. İlgili çalışmada FIFO tasarımına ek olarak

shift register tasarımı da mevcuttur. Alınan veri sinyalleri 16 kez örneklenmiş olup ilgili devrenin tasarımı Verilog dilinde yapılmış ModelSim programında simule edilmiştir [6].

Ming Li FPGA tabanlı dijital ve analog ölçüm alabilen entegre devresi üzerinde çalışmalarda bulunmuştur. Seri olarak haberleşebilen A/D çevirici kontrol tasarımı Verilog donanım tanımlama dili kullanarak FPGA üzerinde gerçeklenmiştir. Simulasyon sonuçları da ModelSim'de yapılmıştır. Yine aynı çalışmada RS232 seri haberleşme protokolünün yanında USB seri haberleşme protokolunun tasarımı da gerçeklenmiştir [3].

1.1. Tezin Amacı, Katkısı ve İçeriği

Bu tezin hedefi bilgisayarın standart I/O seri cihazları fare, modem ve klavye arasında seri asenkron alıcı ve verici (UART) tasarlamaktır. Bir FPGA ile yapabilen prototip UART tasarlamak, test etmek ve FPGA prototipini sistem içinde hatalarını ayıklamaktır. UART teknolojisinin özelliklerini analiz etmek ve diğer mevcut UART teknolojileri karşılaştırmaktır.

Tezin kapsamı içinde VERILOG donanım tanımlama dilini kullanarak XILINX Spartan-6 FPGA Board X-SP6-X9 ortamında bir UART tasarlanmıştır. VERILOG tasarım kodunu sentezlemek için UART algoritması davranışsal seviyede yazılmış ve FPGA cihazına yüklenmiştir. Kodu doğrulamak için VERILOG dilini kullanarak XILINX Spartan-6 FPGA Board X-SP6-X9 ortamı altında çalışan UART çalıştırılmıştır.

Bu tezin diğer çalışmalardan farkı ise FPGA ile mikrodenetleyicinin bilgisayar aracılığıyla seri haberleşmesidir. Bilgisayarın iki usb port çıkışı aynı anda kullanılarak uygulamalarda yaygın olarak kullanılan iki elemanı FPGA ve mikrodenetleyiciyi bir araya getirmesidir.

İkinci bölümde uart, FPGA, mikrodenetleyici ve diğer kullanılan argümanlar hakkında temel bilgiler verilmiştir.

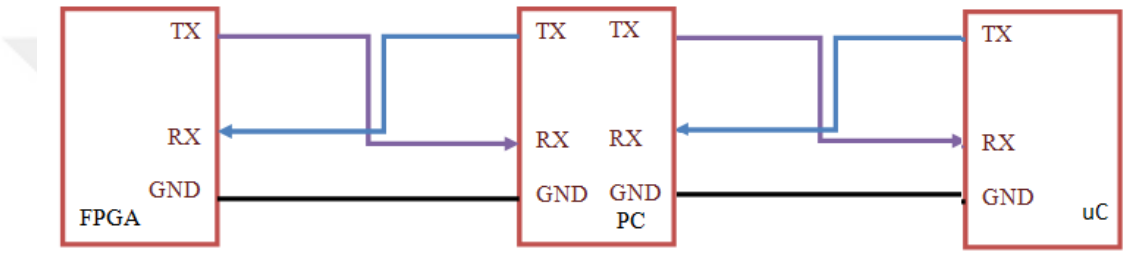
Üçüncü bölümde FPGA ve mikrodenetleyicinin uart çalışması ve yazılan arayüz programı hakkında bilgiler verilmiştir.

Dördüncü bölümde ise FPGA-bilgisayar-mikrodenetleyici seri haberleşmesine örnek bir uygulama anlatılmıştır.

2. TEMEL BİLGİLER

2.1. Evrensel Asenkron Alıcı/Verici (UART)

Basit bir UART sistemi, sadece üç iletken ile doğru, orta hızlı ve çift yönlü iletişim sağlar. Tx (iletilen seri veriler), Rx (alınan seri veriler) ve toprak iletkeninden oluşur. SPI ve I2C gibi diğer protokollerin aksine , kullanıcı UART donanımına gerekli zamanlama bilgilerini verdiği için saat sinyali gerekmez.



Şekil 2.1: UART sistemi bağlantı şekli.

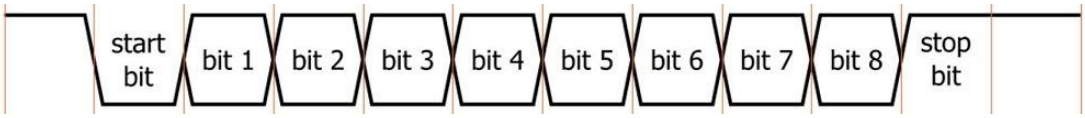
Aslında bir saat sinyali vardır, ancak bir iletişim aracından diğerine iletilmez; daha ziyade, hem alıcı hem de verici, değişen mantık seviyelerinin (Tx tarafında) nasıl oluşturulduğunu ve yorumlanacağını (Rx tarafında) düzenleyen dahili saat sinyallerine sahiptir. Eğer verici ve alıcı farklı veri iletim frekansları için yapılandırılmışsa UART iletişimi çalışmaz. Ayrıca, iç saat sinyalleri, beklenen frekansa göre yeterince doğru; zaman ve sıcaklık üzerinde yeterince kararlı olmalıdır.

• Başlangıç Biti

Bir byte UART iletiminin ilk bitidir. Veri hattının boşta kaldığını belirtir. Boşta kalma durumu genellikle mantıksaldır, bu yüzden başlangıç biti mantıksal düşüktür. Başlangıç biti alıcı ve verici arasındaki iletişimi kolaylaştırdığı, ancak anlamlı verileri aktarmadığı anlamına gelir [10].

• Durma Biti

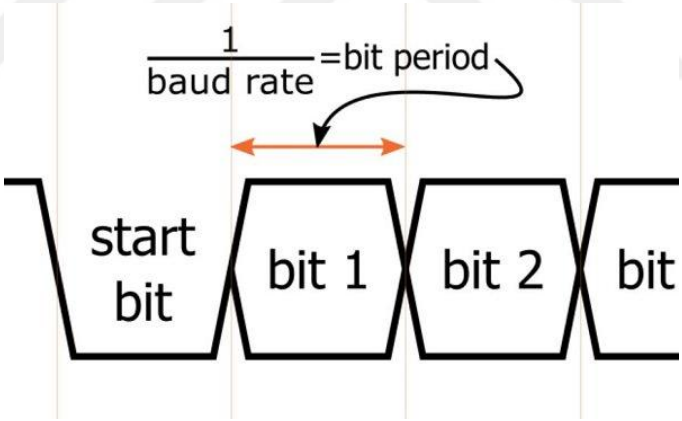
Bir byte UART iletiminin son bitidir. Mantık seviyesi sinyalin rölanti durumuyla aynıdır, yani mantık yüksektir.



Şekil 2.2: RS232 veri çerçeve yapısı.

• Baud Hızı

Verilerin aktarılabileceği yaklaşık hızdır (saniyede bit veya bps). Daha doğru tanımlarsak bir bit dijital veri iletmek için gereken süreye (saniye cinsinden) karşılık gelen frekanstır (bps cinsinden). Örneğin, bir 9600 baud sistemi ile, bir bit 1 / (9600 bps) ≈ 104.2 µs gerektirir. Sistem, saniyede 9600 bit anlamlı veri aktaramaz. Çünkü genel olarak bitler için ek zamana ihtiyaç duyulur ve bazen de bir baytlık iletimler arasındaki gecikmeler olur [10].



Şekil 2.3: Bir bitin peryotu.

• Eşlik Biti

Baytın sonuna eklenen bir hata bulma bitidir. İki çeşiti vardır: Tek eşlik, veri baytı çift sayıda mantık yüksek bit içeriyorsa, eşlik bitinin mantık yüksek olacağı anlamına gelir. Çift eşlikde ise veri baytı tek sayıda mantık yüksek bit içeriyorsa, eşlik bitinin mantık yüksek olacağı anlamına gelir. Ancak iki ve daha fazla bit bozulmuşsa bunun tespiti mümkün değildir.

Ayrıca bu kontrol işleminde hangi bitte bozulma meydana geldiği tespit edilemez. Eşlik biti, özellikle 7 bit içeren ASCII karakterlerinde kullanılır [3]. Eğer

hatasız iletişim için ciddi bir ihtiyaç varsa başka kontrol yöntemleri uygulanabilir. Bu çalışmada parity bitinin bu dezavantajlarından dolayı kullanılmamıştır.

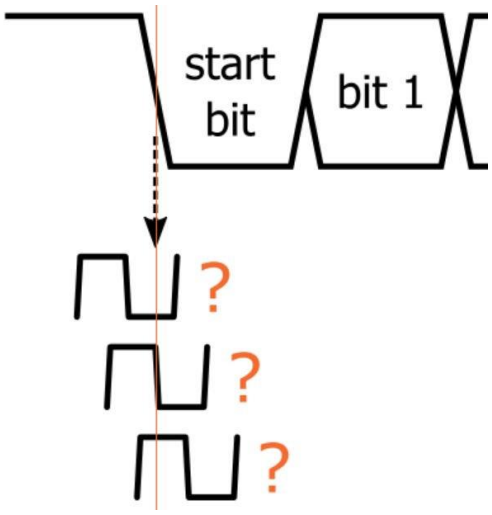
- Senkronizasyon ve Örnekleme

Standart dijital veri, bir tür saat mekanizması olmadan anlamsızdır. Tipik bir veri sinyali, mantıksal düşük ve mantıksal yüksek arasında geçiş yapan basit bir voltajdır. Alıcı, bu mantık durumlarını , sinyali ne zaman örnekleyeceğini bildiyse, dijital verilere doğru şekilde dönüştürebilir . Bu, ayrı bir saat sinyali kullanılarak kolayca yapılabilir. Örneğin; verici, saatin her bir yükselen kenarındaki veri sinyalini günceller ve daha sonra alıcı, yükselen her kenardaki verileri örnekler [3].

Ancak, "evrensel asenkron alıcı/verici " isminin ima ettiği gibi , UART arayüzü Tx ve Rx cihazlarını senkronize etmek için bir saat sinyali kullanmaz. Verici, saat sinyaline bağlı olarak bir bit akımı üretir ve daha sonra alıcının amacı, her bit periyodunun ortasında gelen verileri örneklemek için dahili saat sinyalini kullanmaktır. Bit periyodunun ortasındaki örnekleme gerekli değildir, ancak optimaldir, çünkü bit periyodunun başlangıcına veya sonuna daha yakın örnekleme, sistemi alıcı ve verici arasındaki saat frekansı farklılıklarına karşı daha az dayanıklı hale getirir [3].

Alıcı ve vericideki seans başlangıç bitleri tamamen kodu yazan kişiye aittir. Bazı programcılar bitin düşen kenarını, yükselen kenarını veya tam ortasını baz alabilirler.

Bu tezdeki çalışmada alıcı seansı başlangıç bitinin yükselen kenarı ile başlar. Bu kritik senkronizasyon işleminin gerçekleştiği zamandır. Alıcının dahili saati, vericinin dahili saatinden tamamen bağımsızdır. Başka bir deyişle, bu ilk yükselen kenar, alıcının saat döngüsündeki herhangi bir noktaya karşılık gelebilir.



Şekil 2.4: Bir bitin kenar tetiklenmesi.

Alıcı saatinin aktif bir kenarının bit periyodunun ortasına yakın bir yerde gerçekleşmesini sağlamak için, alıcı modülüne gönderilen baud oranı saatinin frekansı (8 veya 16 veya hatta 32'lik bir faktörle) çok daha yüksektir [10].

FPGA aygıtında örnekleme programcının istediği şekilde ayarlanabilir. Mikrodenetleyicide ise donanım olanaklarıyla sınırlıdır. Örnekleme oranı arttıkça hata payı azalmaktadır.

Bir bit periyodunun 16 alıcı saat döngüsüne karşılık geldiğini varsayalım. Bu durumda, senkronizasyon ve örnekleme şu şekilde devam edebilir:

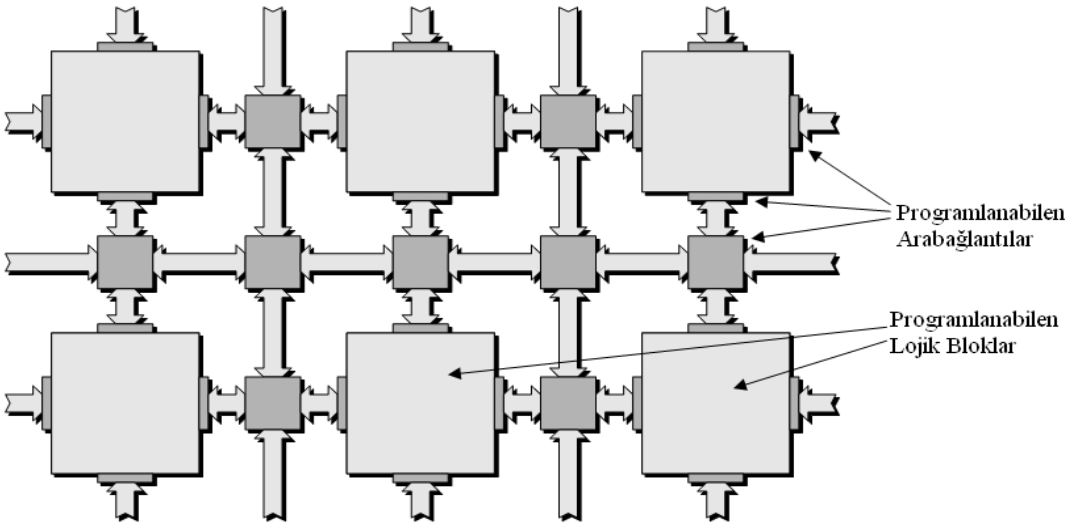
- Alma işlemi, başlangıç bitinin düşen kenarı tarafından başlatılır.
- Alıcı, bit periyodunun ortasına yakın bir örnekleme noktası oluşturmak için 8 saat döngüsünü bekler.
- Alıcı daha sonra ilk veri-bit periyodunun ortasına getiren 16 saat döngüsünü bekler.
- İlk veri biti örneklenir ve alma kaydında saklanır, daha sonra modül ikinci veri biti örneklemeden önce başka bir 16 saat döngüsünü bekler.
- Bu işlem, tüm veri bitlerinin örneklenip depolanmasına kadar tekrarlanır ve daha sonra durdurma bitinin yükselen kenarı, UART arayüzünü boşta durumuna döndürür [11].

2.2. FPGA Tanımı

FPGA, sahada programlanabilir kapı dizisi anlamına gelir. Aslında, bir FPGA ortak işlevleri yerine getiren ve aynı zamanda çok yüksek düzeyde esneklik sunan, birbirine bağlı dijital alt devreler dizisidir. İyi bir isim oldukça bilgilendirici olabilir ve "sahada programlanabilir kapı dizisi" oldukça iyi bir isim olarak kabul edilir. Bir FPGA bir mantık kapıları dizisidir ve onu tasarlayan insanlar tarafından bu dizi programlanabilir veya yapılandıralabilir şeklinde tanımlanabilir [3].

Lojik kapılar (AND, VEYA, XOR, vb.) dijital devrelerin temel yapı taşlarıdır. Bir şekilde birbirine bağlanabilen çok sayıda lojik kapıların yapılandırılabilir olması amaçlanan dijital cihazın oluşmasını sağlar [18].

Bununla birlikte, bir FPGA bireysel Boole kapılarının geniş bir koleksiyonu değildir. Bu, yapılandırılabilir mantık işlevselliği sağlamanın en alt yoludur çünkü ortak işlemlerin sabit modüller olarak daha verimli bir şekilde gerçekleştirilebilmesi gerçeğinden sağlamayacaktır. Aynı ilke, ayrı dijital IC'lerin dünyasında da belirgindir. AND kapılar, OR kapılar ve benzerlerinden oluşan IC'ler satın alınabilir ancak bireysel kapılardan bir shift register oluşturmak istenilmez. Bunun yerine, bir shift register IC satın alınır [15]. Öyleyse bir FPGA, bir kapı dizisinden çok daha fazlasıdır. Çok yüksek düzeyde esneklik sunarken, ortak işlevleri verimli bir şekilde uygulayan, dikkatlice tasarlanmış ve birbirine bağlı dijital alt devreler dizisidir. Dijital alt devrelere yapılandırılabilir mantık blokları (CLB'ler) denir [3].

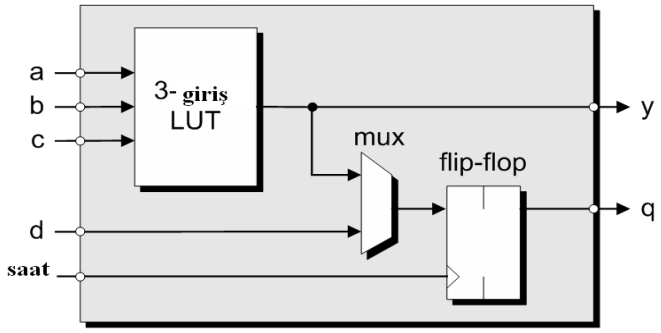


Şekil 2.5: FPGA yapısı.

CLB'lerin birbirleriyle ve harici devrelerle etkileşime girmeleri gerekir. Bu amaçlar için FPGA, programlanabilir ara bağlantı ve I/O blokları matrisini kullanır. FPGA'in programı, CLB'lerin işlevselliğini etkileyen ve bağlantı yollarını kuran anahtarları kontrol eden SRAM hücrelerinde depolanır.

CLB'lerin look up tablolarını, depolama elemanlarını (flip-flop'lar veya register) ve CLB'nin Boolean, veri depolama ve aritmetik işlemler gerçekleştirmesini sağlayan çoklayıcıları içermesidir .

Bir I/O bloğu, CLB'ler ve karttaki diğer bileşenler arasında iletişimi kolaylaştıran çeşitli bileşenlerden oluşur. Bunlar, pull up/down dirençleri, tamponlar ve invertörleri içerir. [20].



Şekil 2.6: FPGA mantık bloğunun yapısı.

Aslında, FPGA uygulamasının genel olarak bir mikrodenetleyicinin programlanmasından daha zor olduğu düşünülmektedir. Bununla birlikte, FPGA geliştirme CLB işlevselliği veya iç ara bağlantıların özenli bir şekilde düzenlenmesi hakkında tam bir bilgi gerektirmez.

Aslında, bir FPGA'yi bağımsız bir bileşen olarak sunmak biraz yanıltıcıdır. FPGA'ler her zaman, bir donanım tasarımını ara bağlantıların ve CLB'lerin davranışını belirleyen programlama bitlerine dönüştürmenin karmaşık işlemini gerçekleştiren bir geliştirme yazılımı tarafından desteklenir [9].

FPGA LUT'larını herhangi bir mantıksal işlevi uygulamak için bir ön kaynak olarak kullanır. Bu aslında iki aşamalı bir süreçtir. İlk başta, Boolean işlevini oluşturan girdi değişkenlerinin her bir birleşimi için çıktı değerleri LUT'un SRAM hücrelerinde depolanır. Bundan sonra, kullanıcı tarafından sağlanan giriş değişkenlerinin kombinasyonuna bağlı olarak, uygun bellek biti LUT çıkış piminde

görünecektir. Bunun nedeni, kullanıcı tarafından sağlanan giriş bitlerinin, LUT'lar içinde mevcut olan çoklayıcılar için seçim hattı görevi görmesidir.

Modern FPGA'ler, veri toplamak, ASIC'leri kontrol etmek ve matematiksel işlemler yapmak için mikrodenetleyicileri kullanmaya alışkın olanlar için biraz karmaşık olabilen çok yönlü, yüksek performanslı cihazlardır. İnsanların donanımı tanımlamayı sağlayan diller oluşturdular. Bunlara hardware description languages yani donanım tanımlama dilleri (HDL'ler) denir ve en yaygın olanları VHDL ve Verilog'dur. HDL kodu ve yüksek seviye yazılım programlama dilinde yazılmış kod arasındaki belirgin benzerliğe rağmen, ikisi temelde farklıdır. Yazılım kodu, bir işlem sırasını belirtirken, HDL kodu ise bileşenleri tanıtmak ve ara bağlantılar oluşturmak için metni kullanan bir şemaya benzer. Donanım tanımlama dilleri FPGA'lerda ve paralel programlama uygulamalarında çok kullanılır. Genellikle donanıma dayalı oldukları için donanım açıklamasındaki diğer dil türlerinden oldukça farklıdırlar [5].

Verilog, 80'lerin başında, öncelikle elektronik sistemlerin modellenmesinde kullanılan ilk HDL'lerden biri olarak icat edildi. Dil adı "VERIfication of LOGic"ın kısaltılmış bir versiyonudur. Program, bir sistemi tanımlayabilmek için bir modül hiyerarşisine dayanıyor. C ile programlama konusunda zaten deneyiminiz varsa, Verilog'u öğrenmek daha kolay olabilir.

VHDL, ABD Savunma Bakanlığı tarafından 80'lerin sonunda, başlangıçta ASIC davranışını daha iyi anlamanın bir yolu olarak geliştirilmiştir. Sonunda Ada programlama diline dayanarak bir HDL haline geldi. VHDL endüstriyel uygulamalarda sıklıkla kullanılır [2].

FPGA'in programlama aşamaları:

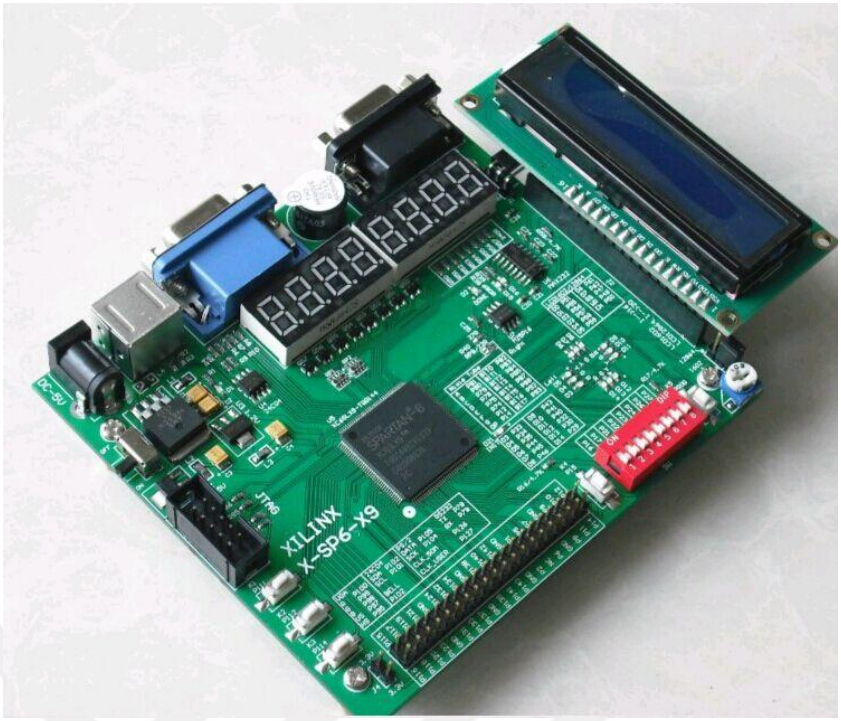
- Sentezleme ile yazılan kod eşdeğer RTL donanıma dönüştürülür.
- Mapping ile mevcut I/O pinleri ile oluşturulan donanım arasında bağlantı kurulur.
- Place and Route işlemi ile programlanabilir lojik bloklar ve ara bağlantılar FPGA içine yerleştirilir.
- En son bit veya hex dosyası oluşturularak FPGA programlanır.

2.2.1. Xilinx

Xilinx'in FPGA portföyü endüstriyel, haberleşme altyapısı, medikal cihaz, otomotive ve tüketici pazarlarının entegrasyon ve performans gereksinimlerini karşılamasını sağlar. Platform ve işlemcinin doğru kombinasyonu ile mühendisler, I/O çevre birimleri, iletişim arayüzleri, gerçek zamanlı çalışmalar ve işletim sistemi desteğinin doğru kombinasyonunu içeren mimari zorlukları karşılamak için bir çözüm sağlar. [25].

Spartan6 ailesi, büyük hacimli uygulamalar için en düşük toplam maliyeti olan lider sistem entegrasyon yetenekleri sunar. On üç üyeli aile, önceki Spartanlı ailelerin güç tüketiminin yarısı ve daha hızlı, daha kapsamlı bağlantı ile, 3.840 ila 147.443 mantık hücreleri arasında genişletilmiş yoğunluklar sunar. Optimum maliyet, güç ve performans dengesi sağlayan 45 nm'lik düşük güçlü bir bakır işleme teknolojisi üzerine inşa edilmiş Spartan-6 ailesi, yeni, daha verimli, çift kayıtlı 6 girişli bir arama tablosu (LUT) mantığı ve yerleşik sistem düzeyinde blokların zengin bir seçim sunar. Bunlara 18 Kb (2x9 Kb) blok RAM, ikinci nesil DSP48A1 dilimleri, SDRAM bellek denetleyicileri, gelişmiş karma modlu saat yönetim blokları, SelectIO™ teknolojisi, güçlendirilmiş yüksek hızlı seri alıcı blokları, PCI Express® uyumlu uç nokta blokları, gelişmiş sistem düzeyinde güç yönetimi modları, otomatik algılama yapılandırma seçenekleri ve gelişmiş AES ve Cihaz DNA koruması ile IP güvenliğine sahiptirler. Bu özellikler, benzeri görülmemiş kullanım kolaylığına sahip özel ASIC ürünlerine düşük maliyetli programlanabilir bir alternatif sunar. Spartan6 FPGA'lar için en iyi çözümü sunar. Yüksek hacimli mantık tasarımları, tüketiciye yönelik DSP tasarımları ve maliyete duyarlı gömülü uygulamalardır. Spartan6 FPGA'ler, tasarımcıların gelişim süreci başlar başlamaz yeniliğe odaklanmalarını sağlayan entegre yazılım ve donanım bileşenleri sunan hedeflenen tasarım platformları için programlanabilir silikon temellidir.

Xilinx, FPGA'ler için geliştirme çabalarını basitleştirmek ve sistem bütçelerini en aza indirgemek için araç paketi ve FPGA platformunda gelişmeler sunar. Xilinx'in birkaç farklı geliştirme ortamı vardır. Xilinx araçları çoğu geliştirme kartını destekleyen ücretsiz bir web sürümü sunar [20]. Xilinx, CPLD ve FPGA geliştirme kitleri için USB2 indirme kablolarının yanı sıra gerektiğinde cihazlara doğrudan bağlantı için JTAG kabloları da sağlar.



Şekil 2.7: XILINX Spartan-6 FPGA Board X-SP6-X9.

FPGA Board özellikleri: [11].

i) Xilinx Spartan-6 xc6slx9-tqg144 chip
ii) JTAG ara bağlantı
iii) 50MHZ kristal ve değiştirilebilir kristal
iv) Onboard 5 key input,4 keys can proggramer
v) M25P16 16M bit spi flash.configer FPGA
vi) AT24C04 I2C
vii) 1 X 8 swicth
viii) On-board 12-bit LED
ix) On-board 8-bit 7segment display
x) Ses deneyleri için buzzer
xi) RS232 MAX232 seri haberleşme noktası
xii) PS/2 klavye bağlantı noktası
xiii) 1602LCD karakter LCD arabağlantı
xiv) VGA display
xv) 28 PIN I/O to 2.54

2.3. Mikrodenetleyici

Mikrodenetleyici, belirli bir görev veya uygulamayı gerçekleştirmek için tasarlanmış bir çiptir. Mikrodenetleyici CPU (mikroişlemci), RAM, ROM, I/O portları, zamanlayıcılar, sayıcılar içerir. Entegre bir mikrodenetleyicinin küçültülmüş boyutu ve maliyeti, birçok otomatik cihaz ve işlemin kontrolü için ekonomik bir çözümdür [12].

En yaygın giriş ve çıkış aygıtları; okuma sıcaklıkları, nem ve ışık, LED'ler, radyo frekans aygıtları, röleler, solenoitler, küçük LCD'ler ve anahtarlar için veri sensörlerini içerir [24].

Bir mikrodenetleyici kullanarak Microsoft Excel gibi bir yazılım uygulamasını çalıştıramayacağı doğru olsa da, bu bilgisayarlar küçük ama güçlüdür. Belirli cihazlar arasındaki kontrol yönleri için mükemmel şekilde tasarlanmıştır. Dünya şimdi bu teknolojiye çok güveniyor ve gelecekte de bunu yapmaya devam edecektir [17].

2.3.1. PIC

Microchip Teknolojisi tarafından yapılan PIC Mikrodenetleyiciler ailesine genel olarak verilen addır. Microelectrics Division of General Instruments tarafından geliştirilen PIC1650'yi temel almaktadır. Arayüz kontrol cihazı (Peripheral Interface Controller) olarak da bilinir. 8, 16 veya 32 bit veri belleği ile birlikte gelir. Mevcut PIC modelleri, ROM veya EPROM ile sınırlı kaldıkları günlerden çok uzakta olan flash belleği kullanır.

PIC'in yeniden programlanabilir hafızası ve nispeten düşük maliyetli olması, onları elektronik geliştiricileri ve hobileri için popüler bir seçenek haline getirmektedir. Bu popülerlik, popülerliğini arttırmaya devam eden çok sayıda belge, not, eğitim ve düşük maliyetli geliştirme araçlarıyla sonuçlanmıştır [23].

PIC mikrodenetleyicisini programlamak için CCS C derleyicisi kullanıldı. Bu derleyici ücretli olmasına karşın sınırlı kullanım olanağı sağlayan ücretsiz sürümü bu çalışmada istenilen işler için yeterlidir. CCS C derleyicisi C ve Assembly dilinde program yazmayı desteklemektedir.

2.3.2. Arduino

Arduino, elektronik projeler yapmak için kullanılan açık kaynaklı bir platformdur. Arduino, üzerinde Atmel firmasının ürettiği mikrodenetleyici bulunan fiziksel bir programlanabilir devre kartı ve bilgisayarda yazılan kodu yükleme kartı olmadan yüklemek için mikrodenetleyicide yüklü olan bootloaderdan oluşur.

Arduino platformu, elektronikde yeni başlayan insanlar arasında oldukça popüler hale geldi. Daha önceki programlanabilir devre kartlarının çoğundan farklı olarak, Arduino, karta yeni kod yüklemek için ayrı bir donanıma ihtiyaç duymaz, sadece bir USB kablosu kullanılabilir. Ek olarak, Arduino IDE basitleştirilmiş bir C++ sürümü kullanır ve programlamayı öğrenmeyi kolaylaştırır. Son olarak, Arduino, mikrodenetleyicinin hazır fonksiyonlarına internet ortamında açık kaynak kodlu olarak erişilebilirlik sağlar [22]. Fakat bootloader ramda çok geniş yer tuttuğu için büyük programlara yer kalmaz.

2.4. C# Dili

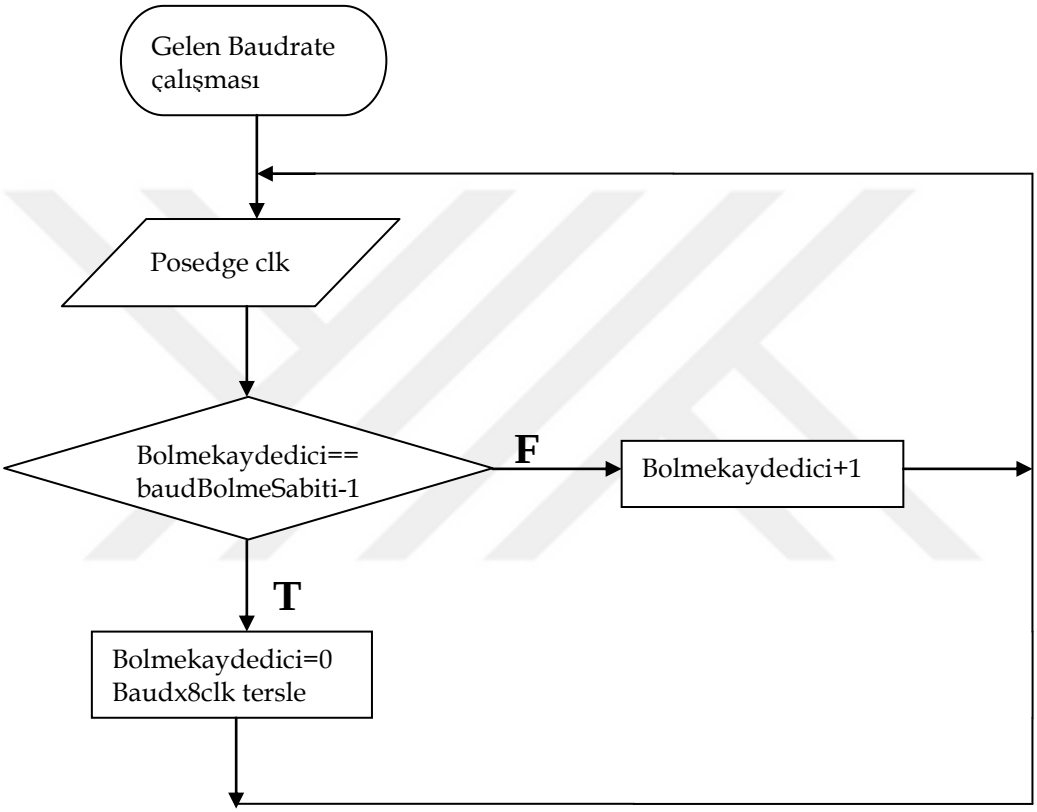
C# geliştirilen modern, genel amaçlı, nesne yönelimli bir programlama dilidir. Microsoft, Uluslararası Standartlar Örgütü (ISO) ve Avrupa Bilgisayar Üreticileri Birliği (ECMA) tarafından onaylandı. C# .Net'in gelişimi sırasında Anders Hejlsberg ve ekibi tarafından geliştirilmiştir [24].

C#, ortak dil altyapısı (CLI) için tasarlanmıştır. Ortamı farklı bilgisayar platformlarında ve mimarilerde çeşitli üst düzey dillerin kullanımlarına izin veren kodlar için yapılmıştır. Her ne kadar C# yapısı geleneksel üst seviye dilleri yakından takip etse de C ve C++ ve nesne yönelimli bir programlama dilidir. Modern, genel amaçlı, nesne odaklı, bileşen odaklı, öğrenmesi kolay, çeşitli bilgisayar platformlarında derlenebilmesi, verimli programlar üretmesi, yapılandırılmış bir programlama dil olması, Framework .Net Framework'ün bir parçası olması C# 'ı yaygın olarak kullanılan bir profesyonel dil yapar [19].

3. UART SİSTEMİNİN ANALİZİ

3.1. FPGA Seri Haberleşmesi

3.1.1. Gelen Baudrate Üretecinin Algoritması



Şekil 3.1: Gelen baudrate üretecinin algoritması.

Baudrate birim zaman içinde yollanan yada alınan bit sayısını belirleyen bir haberleşme hızı birimidir. 9600 baudrate denildiğinde bir saniye içinde 9600 tane bit gönderilmiş yada alınmış anlamına gelmektedir. Spartan-6 kartı üzerinde 50MHz değerinde bir kristal bulunduğu için programda baudrate oranı aşağıdaki gibi hesaplanır:

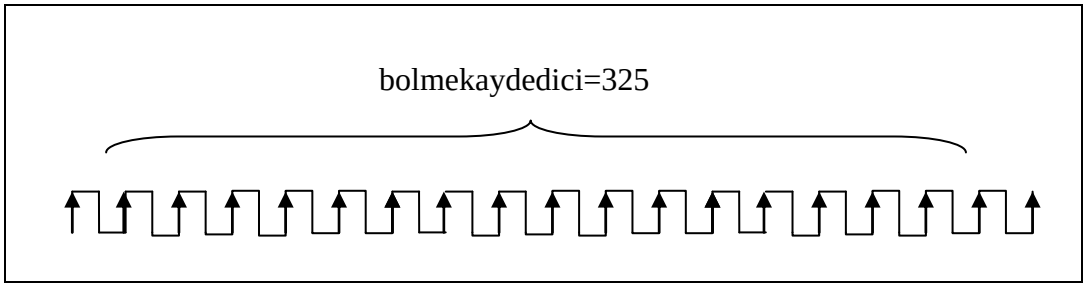
FPGA kristal = 50.000.000 **(**50Mhz**)**

baudBolmeSabiti = 50.000.000 / **(**BaudRate x örnekleme oranı x 2**)** [3] (3.1)

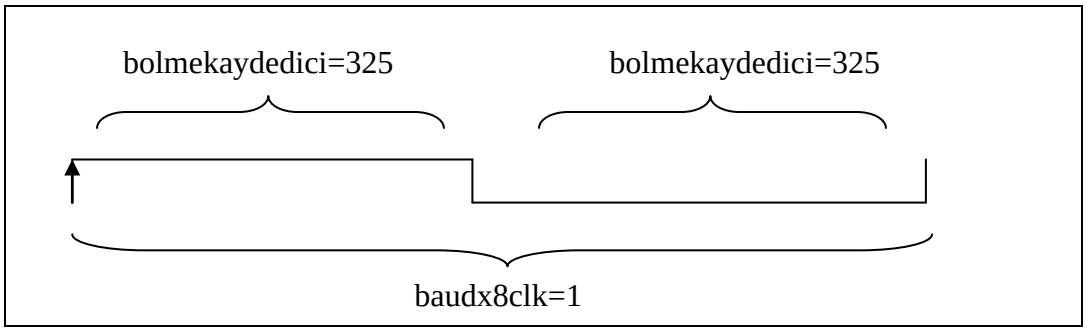
= 50.000.000 / **(** 9600 x 8 x 2 **)** = 325

Bu formülü açıklamak gerekirse 9600 baudrate hızına sahip bir sistemde 1 saniyede 9600 tane bit gönderiliyorsa 1 bit kaç saniyede gönderilirin cevabı bulunur. Bu cevap bize 1/9600=0,0001041 saniyeyi verir. FPGA osilatörü (50MHz) bir saniyede 50 000 000 tane kare dalga üretir. 1 kare dalga için geçen süre 1/50 000 000 ‘ dan 0,00000002 saniyeyi verir. 1 bitin geçmesi için gerekli olan kare dalga sayısı 0,0001041/ 0,00000002 oranından 5205 bulunur. Yani 1 bit için seri iletişimde FPGA 5205 tane kare dalga üretmesi gerekir. UART asenkron olduğundan, yalnızca aşırı örnekleme ile örneklenebilir. 5205 tane kare dalga üzerinde 8x, 16x ve 32x aşırı örnekleme alınabilir. Bu uygulamada yeterli görülerek 8x aşırı örneklemesi alınmıştır. Her kare dalganın yükselen kenarı baz alınarak sistem çalıştırılmıştır. Bir peryot yükselen ve düşen kenar içerdiği için 2x8=16 tane kare dalgadan oluşur. 5205/16 oranından her 325 kare dalgada bir sistem kontrol edilerek çalıştırılmıştır.

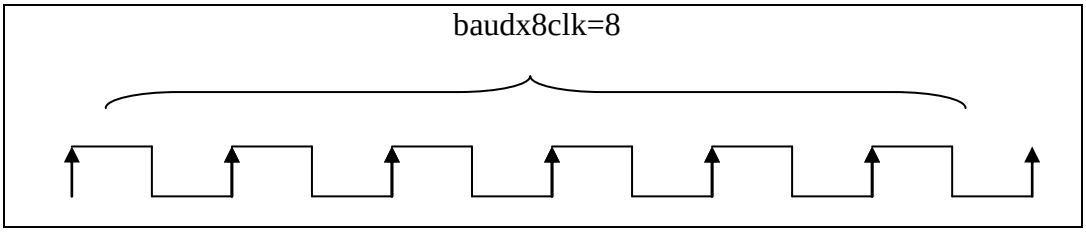
Algoritmanın açıklaması ise posedge clk ile FPGA osilatörünün ürettiği kare dalganın her yükselen kenarında programın döngüye girmesini sağlar. Kare dalga sayısı sıfırdan saymaya başladığı için baudBolmeSabiti’nin (325) bir eksiğine kadar bolmekaydedici’si arttırılır. Bolmekaydedici’si 324 olduğunda kendisi sıfırlanır ve baudx8clk ise 1 ise sıfır; 0 ise 1 seviyesine getirilir [21].



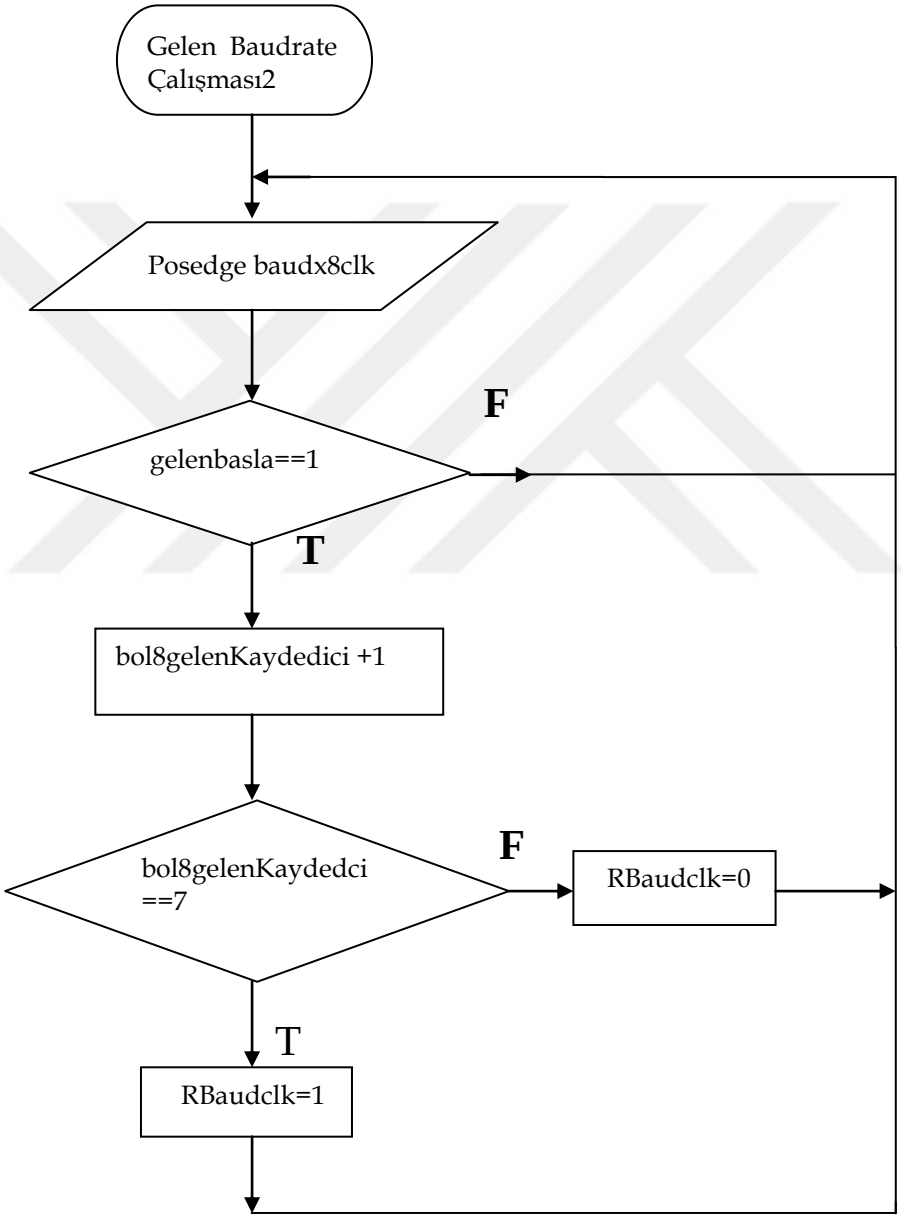
Şekil 3.2: Bölmekaydedici değişkeninin kare dalga diyagramı.



Şekil 3.3: baudx8clk değişkeninin bir peryotu.



Şekil 3.4: baudx8clk değişkeninin kare dalga diyagramı.



Şekil 3.5: Gelen baudrate çalışması üretecinin algoritması.

Yukarıdaki algoritma oluşan her baudx8clk sinyalinin yükselen kenarında programa girer. Program ne zaman bilgi geldiğini anlarsa gelenbasla değişkenini 1 yapar. Gelenbasla değişkeni 0 ise program hiçbir şey yapmadan döngünün başına döner. Gelenbasla değişkeni 1 ise her 650 yükselen kenarda bol8gelenKaydedici değişkeni 1 artar. bol8gelenKaydedici değişkeni 7'ye eşit ise RBaudclk değişkeni 1'e eşitlenir. Aksi halde 0'a eşitlenir. RBaudclk değişkenin 1'e eşit olması bir bitlik sürenin oluştuğu anlamına gelir. 7' e kadar sayınca bir 2*325 kadar kare dalga gelince toplam 5205 tane peryot oluşmuş olur.

3.1.2. Bilgisayardan Gelen Bilgiyi Karşılama Algoritması

Programın bilgisayardan gelen bilgiyi karşılama bölümü her posedge baudx8clk yani 650 tane yükselen kenar geldiğinde döngü içine girer. FPGA karşıdan gelen bilgilerin karşılandığı ucu Rveri olarak atadık. Gelen bilgi Rveri ucundan okunarak döngü içerisinde RVeriKaydedici1 değişkenine atanır. Tıkanmasız (non-blocking) işleci "<=" RVeriKaydedici1 değişkeninin eski değerini RveriKaydedici2 değişkenine atar. İlk durumda Rdurumu değişkeni 0 olduğu için buradan bilginin karşıdan gönderilmeye başlandığının kontrolü için önce RbaslaGecici değişkenin içi kontrol edilir. Bu yapı ilk kontrolde false verir. Sonra !RVeriKaydedici1&&RVeriKaydedici2 mantığı kontrol dilir. Eğer bilgi karşıdan gelmeye başlamışsa hat önce 1 sonra 0 olmuştur. Bu durumda RVeriKaydedici1=0 ve RVeriKaydedici2=1 olur. RveriKaydedici1 tersi alındığı için AND operatörünün sağ ve sol tarafı true verir. Bu şartlı yapı altta RbaslaGecici değişkenini 1 yapar. Tekrar döngünün başına dönülür. Bu sefer RbaslaGecici=1 olduğu için bu şartlı yapı çalışır. Bu şartlı yapı bilginin karşıdan gönderilmeye başladığının (RBasla<=1) yani baudrate oranın 5205 kontrol edilmeye başlandığının, bundan sonra gelecek 9 bitle döngünün tamamlanacağını (RBaslaGecici<=0) ve bundan sonra ilk gelecek bitin ilk anlamlı bit olduğunu belirtir (RDurumu +1).

Bir sonraki posedge baudx8clk olduğunda Rdurumu içeriği bir arttırıldığı için ikinci şartlı yapı (RDurumu>=1&&RDurumu<=8) içine girer. Bu yapı içine girildiğinde hemen rBaudclk==1 şartı yani bir bit için gerekli zaman geçmesi için kontrol yapılır. Bit için gerekli zaman geçtikten sonra girişten alınan bit RveriTampon adlandırdığımız dizinin en sol yani 7. bitine atama yapar (

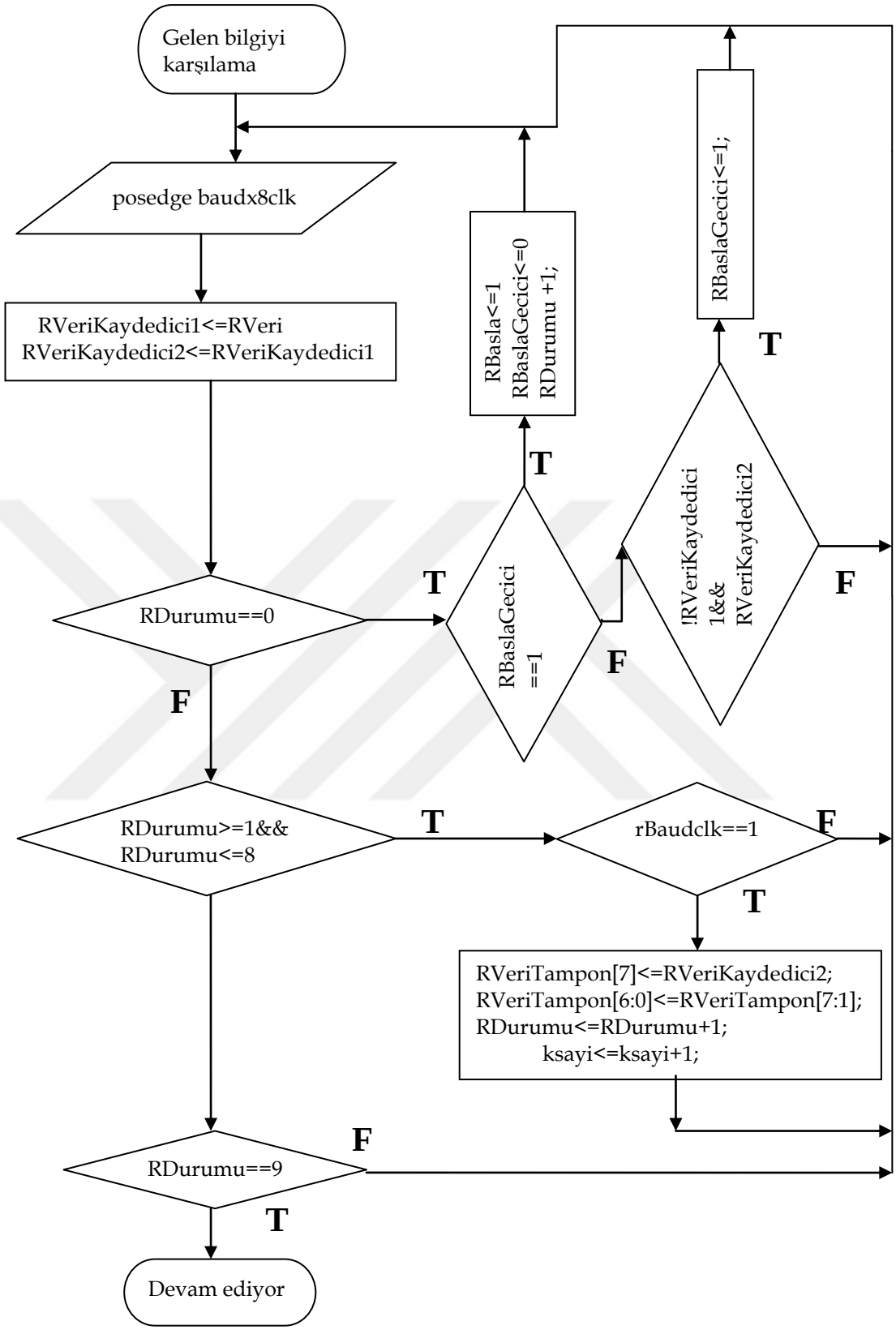
RVeriTampon[7]<=RVeriKaydedici2). Aynı anda bu dizinin sol taraftaki bitleri bir sağa kaydırılır (RVeriTampon[6:0]<=RVeriTampon[7:1]). Kaç tane bit geldiği kaydedilir (RDurumu<=RDurumu+1). Ayrıca kaç tane rakamın geldiği tespit edilir (ksayi<=ksayi+1).

RDurumu==9 şartı sağlandığında bir bitlik süre geçtikten sonra bir baytlık yani toplam 10 bitin geldiği belirtilir (RDurumu<=0). Bayt alma işleminin bittiğinin yeni bayt almaya hazır olduğunu belirtir (RBasla<=0). ksayi==8 şartı sağlanırsa ilk gelen bayt birinci sayının onlar basamağı olarak atanır (bosayi[7:0]<=RVeriTampon[7:0]). İkinci sayı displayde bir süre gösterilmesi için ayarlanan sayaç sıfırlanarak yeni ikinci sayı gelmesi beklenir (baslatbekle<=0).

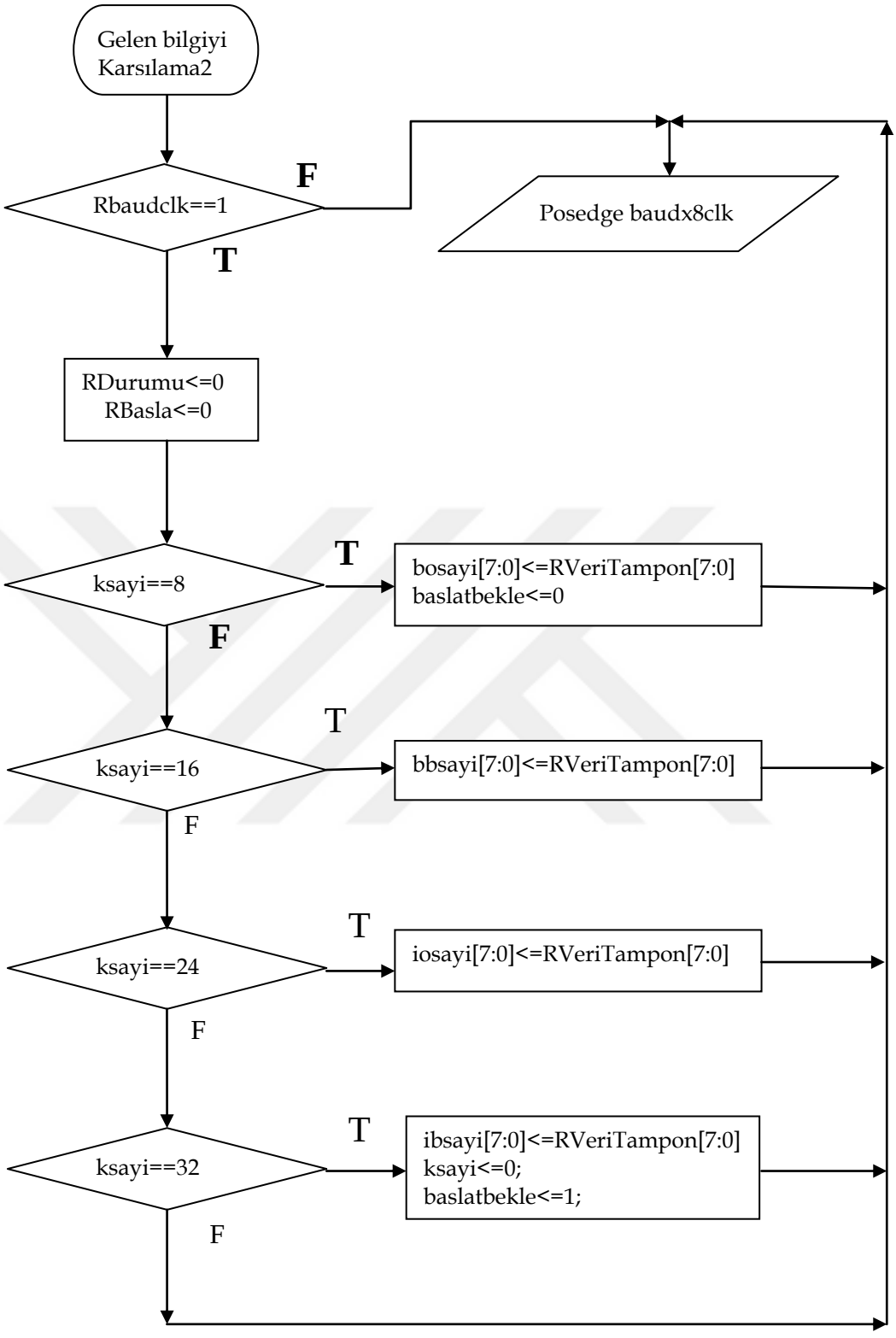
ksayi==16 şartı sağlandığında gelen bayt birinci sayının birler basamağı olarak atanır (bbsayi[7:0]<=RVeriTampon[7:0]).

ksayi==24 şartı sağlandığında gelen bayt ikinci sayının onlar basamağı olarak atanır (iosayi[7:0]<=RVeriTampon[7:0]).

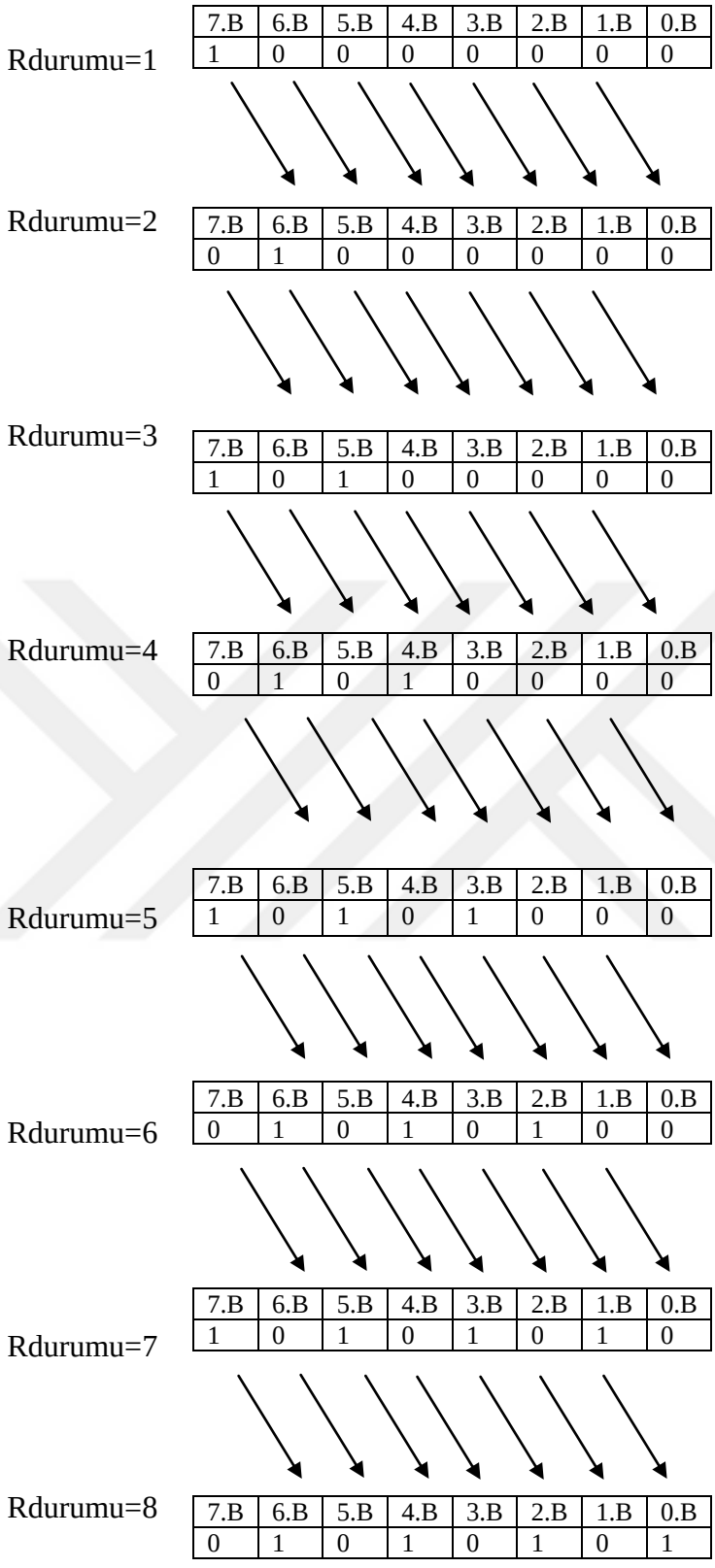
ksayi==32 şartı sağlandığında gelen bayt ikinci sayının birler basamağı olarak atanır (iosayi[7:0]<=RVeriTampon[7:0]). İki basamaklı iki sayının geldiği için yeni gelecek sayılar için sıfırlanır (ksayi<=0). İkinci sayı yeni alındığının bir süre displayde gösterilme ve toplam sonucun displayde gecikmeli gösterilmesi için sayaç kurulmuştur (gecikmebaslatbekle<=1).



Şekil 3.6: Bilgisayardan gelen bilgiyi karşılama algoritması.



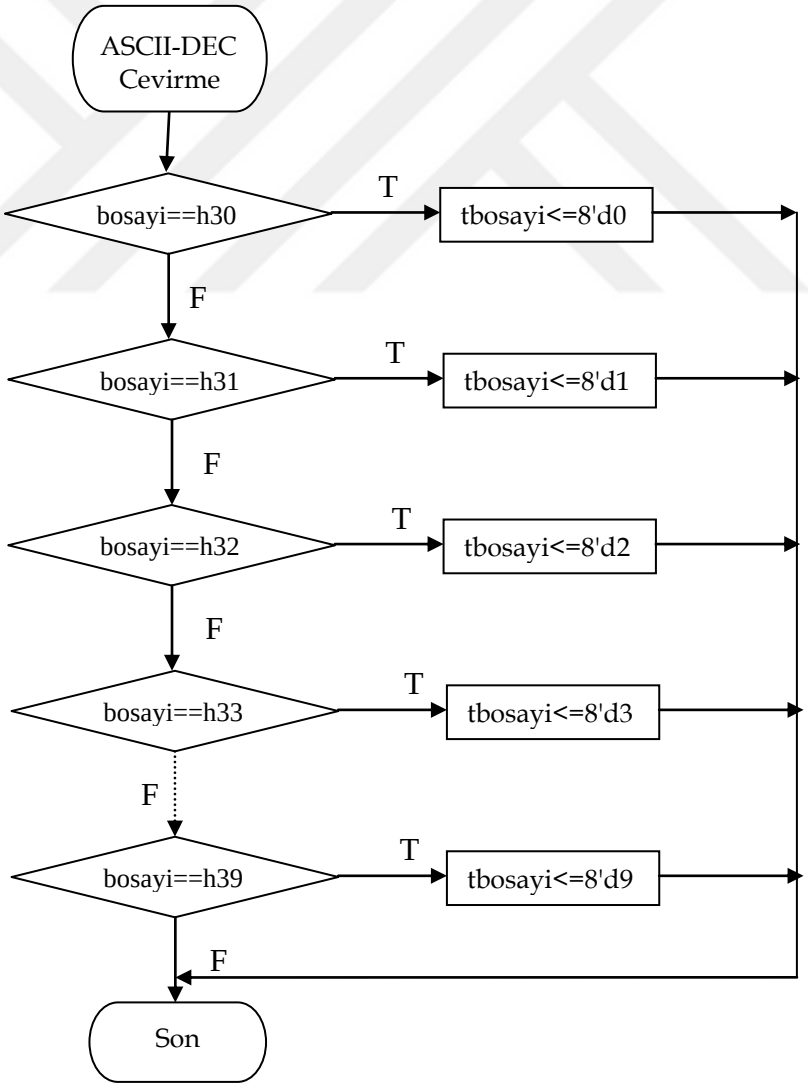
Şekil 3.6: Devam.



Şekil 3.7: $(55)_{16} = (01010101)_2$ sayısının bitlere yerleşimi.

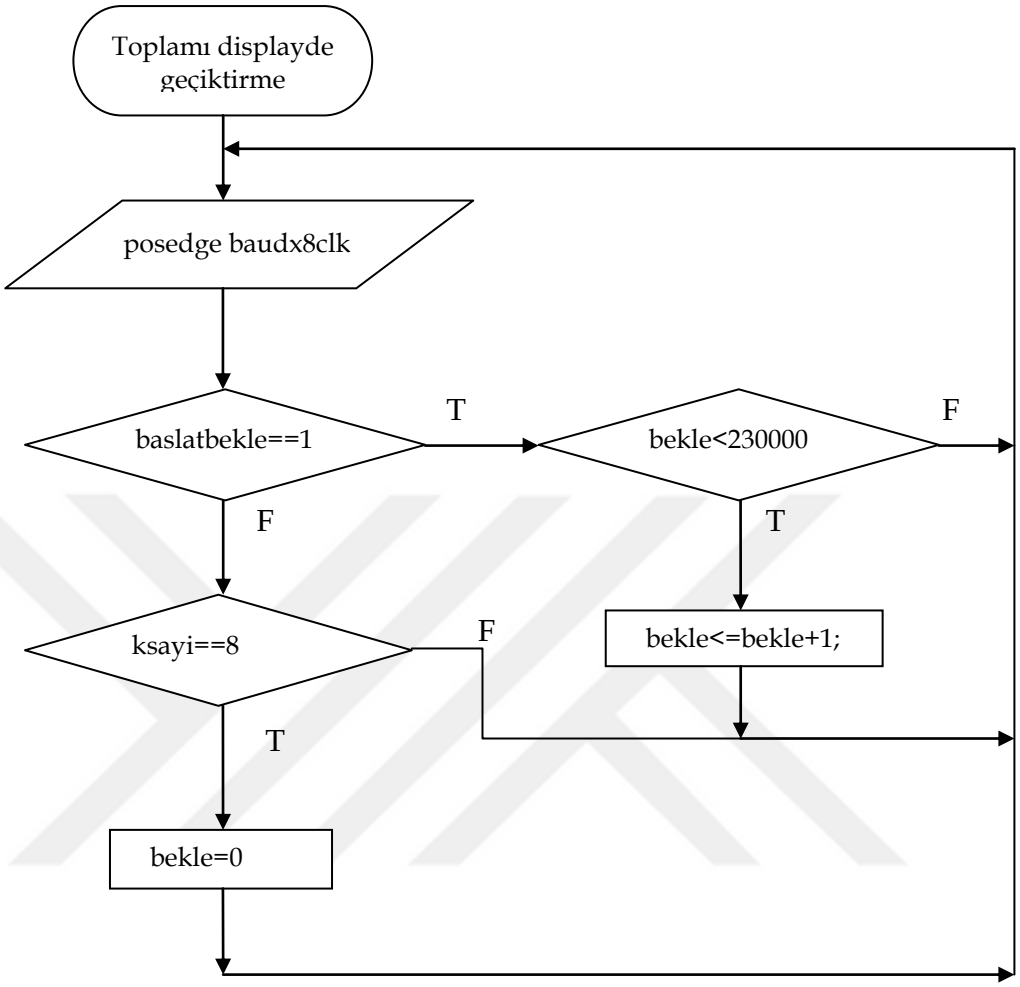
3.1.3. ASCII Kodun Decimal Koda Çevirme

Bilgisayardan gönderilen datalar FPGA aygıtına ASCII kodunda gönderilir. FPGA gelen datalarla matematiksel işlem yapabilmesi için decimal koda çevirmek gerekir. Bilgisayardan gönderilen ASCII kodu hexadecimal şekilde yazılır. Örneğin; bosayi değişkeni hexadecimal 30 sayısına eşit ise bu sayının decimal kodu başka bir değişkene atanır (tbosayi<=8'd0). bosayi değişkeni hexadecimal 30 sayısına eşit değilse diğer dokuz ihtimal sırasıyla denenir. Gelen iki basamaklı sayının rakamları onluk sayı sistemine çevrildikten sonra FIFO mantığına göre birinci ve üçüncü gelen rakamı 10 ile ikinci ve dördüncü sayıları 1 ile çarpıp hepsi toplanır. Böylece gelen iki sayının toplamı bulunur (toplam<=10*tbosayi+tbbsayi+10*tiosayi+tibsayi).



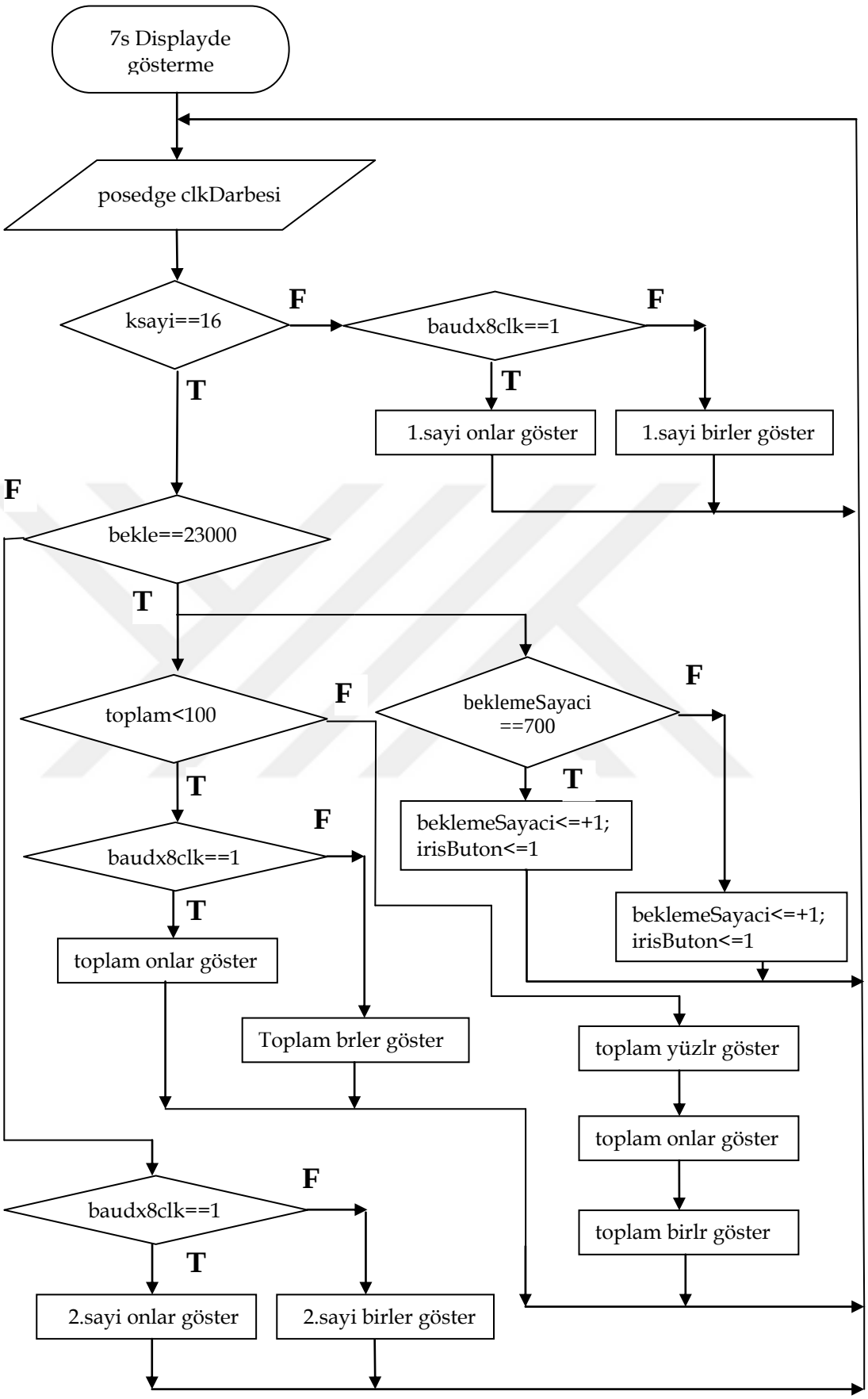
Şekil 3.8: ASCII kodun decimal koda çevirme algoritması.

3.1.4. Sonucun Yedi Segment Displayde Gösterilmesi



Şekil 3.9: Toplamı displayde gösterilmesinin geçiktirme algoritması.

İkinci sayı FPGA elemanına gönderildiğinde mikrosaniyeler içinde birinci sayı ile toplanıp sonuç displayde gösterilecektir. Fakat ikinci sayı displayde neredeyse hiç gözükmeyecektir. İkinci sayının displayde üç saniye gözükmesi için basit bir gecikme programı yazılmıştır. Program her 650 kare dalgada bir döngü içine girer (posedge baudx8clk). İkinci sayının son basamağı olan birler basamağı alındığı zaman baslatbekle değişkenine bir atılır. baslatbekle değişkeni setlenmişse bekle değişkeni 230000 kere arttırılır (bekle<=bekle+1). bekle değişkeni yeni gelen birinci sayının onlar basamağında sıfırlanır (bekle=0).



Şekil 3.10: Sonucun yedi segment displayde gösterilmesi algoritması.

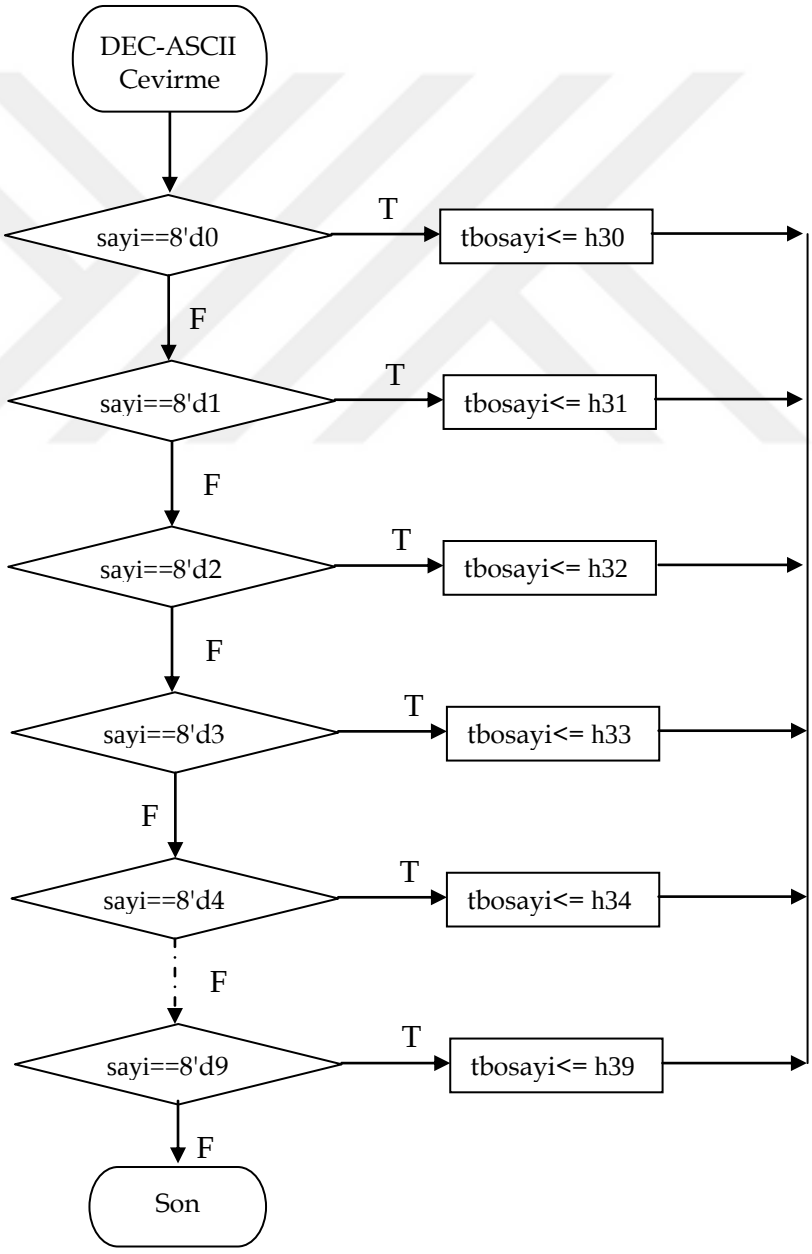
Programın bu kısmı da her yükselen kenarda döngü içine girer. İlk gelen iki basamaklı sayı sonucunda gelen bit sayısı saydırıldığı için ksayi==16 durumu gerçekleşir. 325 tane yükselen kenarda onlar basmağı, ikinci 325 tane yükselen kenarda ise birler basamağı 7segment displayde gösterilecek şekilde ikinci sayı gelene kadar tekrar eder.

İkinci sayı geldiğinde baslatbekle değişkeni set edildiğinden dolayı 3 saniyelik bir gecikme oluşturulmuştur. Bu gecikmeden dolayı ikinci sayı 7segment displayde 3 saniye gözükür ve daha sonra iki sayının toplamı gözükür. Bu gecikme programı ikinci sayının birler basamağı geldiğinde baslatbekle değişkenine bir atılarak başlatılır. Bu gecikme kısmı her yükselen baudx8clk sinyalinde baslatbekle değişkeni bire eşit olduğu sürece bekle değişkeni 230000'e eşit olana kadar arttırılır. bekle değişkeni 230000'e eşit olduğunda programda iki sayının toplamı gösterilir.

Bu kısımda önce gelen iki sayının toplamının iki veya üç basmaklı olduğu tespit edilir. Toplam sayı 100'den küçük olduğu tespit edildikten sonra toplam sayı 10'a bölünerek onlar basamağı, mod10 ile birler basamağı bulunur. 325 tane yükselen kenarda onlar basmağı, ikinci 325 tane yükselen kenarda ise birler basamağı 7segment displayde gösterilecek şekilde yeni sayı gelene kadar tekrar eder. Eğer iki sayının toplamı üç basmaklı yani 99'dan büyük ise üç basamaklı sayının displayde basamak basamak sürülebilmesi için küçük program parçası yazılması gerekir. Bu program her yükselen baudx8clk değişkeninde aktif olur. Aktif olduğunda kbasamak değişkeni 1 arttırılır. Sıfıra eşit olduğunda birler basamağı, bire eşit olduğunda onlar basamağı, diğer durumlarda ise yüzler basamağı gösterilir. İki bitlik tanımlandığından dolayı üçten sonra sıfırlanır. Gkarakter değişkeni iki basamaklı ise içine 2; üç basmaklı ise 3 değeri atanır. Bu değişken gönderme kısmında karşı tarafa kaç basamaklı sayı gönderileceğini belirtir. Ayrıca bu kısımda gönderme kısmının başlaması için gerekli değişkenler set edilir (irisButon<=1). Sadece bir kere toplam sayı gönderilebilmesi için tanımlanan beklemeSayacı sadece 700 sayısına eşit olduğunda bu değişken bire eşit diğer durumlarda sıfıra eşittir.

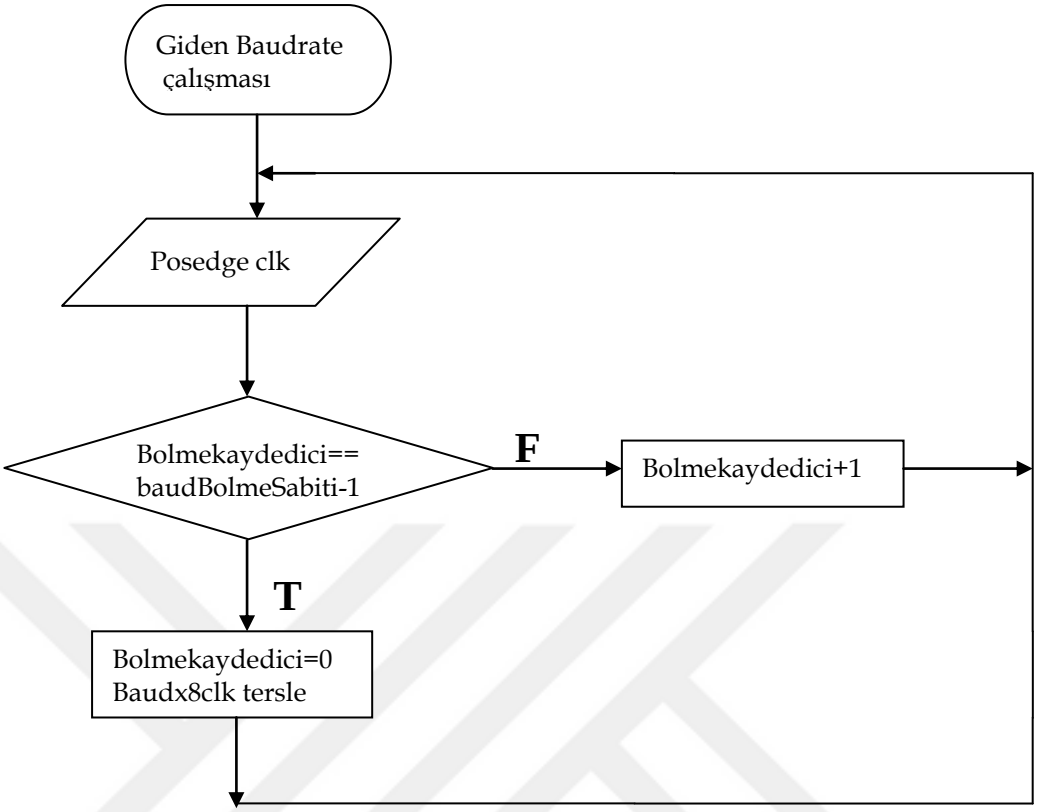
3.1.5. Decimal Kodun ASCII Koda Çevirme

FPGA'de işlenen datalar decimal kodda olduğundan bilgisayara gönderirken ASCII kodunda gönderilir [6]. Bilgisayar gelen datalarla işlem yapabilmesi için ASCII koda çevirmek gerekir. FPGA'den gönderilen data decimal kod şekilde yazılır. Örneğin; sayi değişkeni decimal 0 sayısına eşit ise bu sayının ASCII kodu başka bir değişkene atanır (sayi<= h30). sayi değişkeni decimal 0 sayısına eşit değilse diğer dokuz ihtimal sırasıyla denenir.



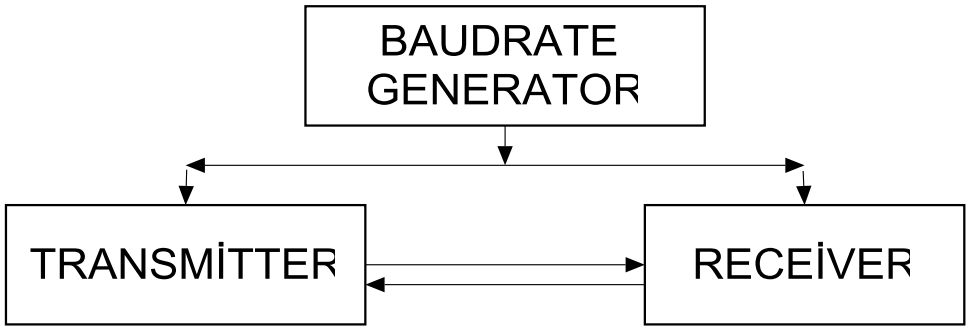
Şekil 3.11: Decimal kodun ASCII koda çevirme algoritması.

3.1.6. Giden Baudrate Üretecinin Algoritması

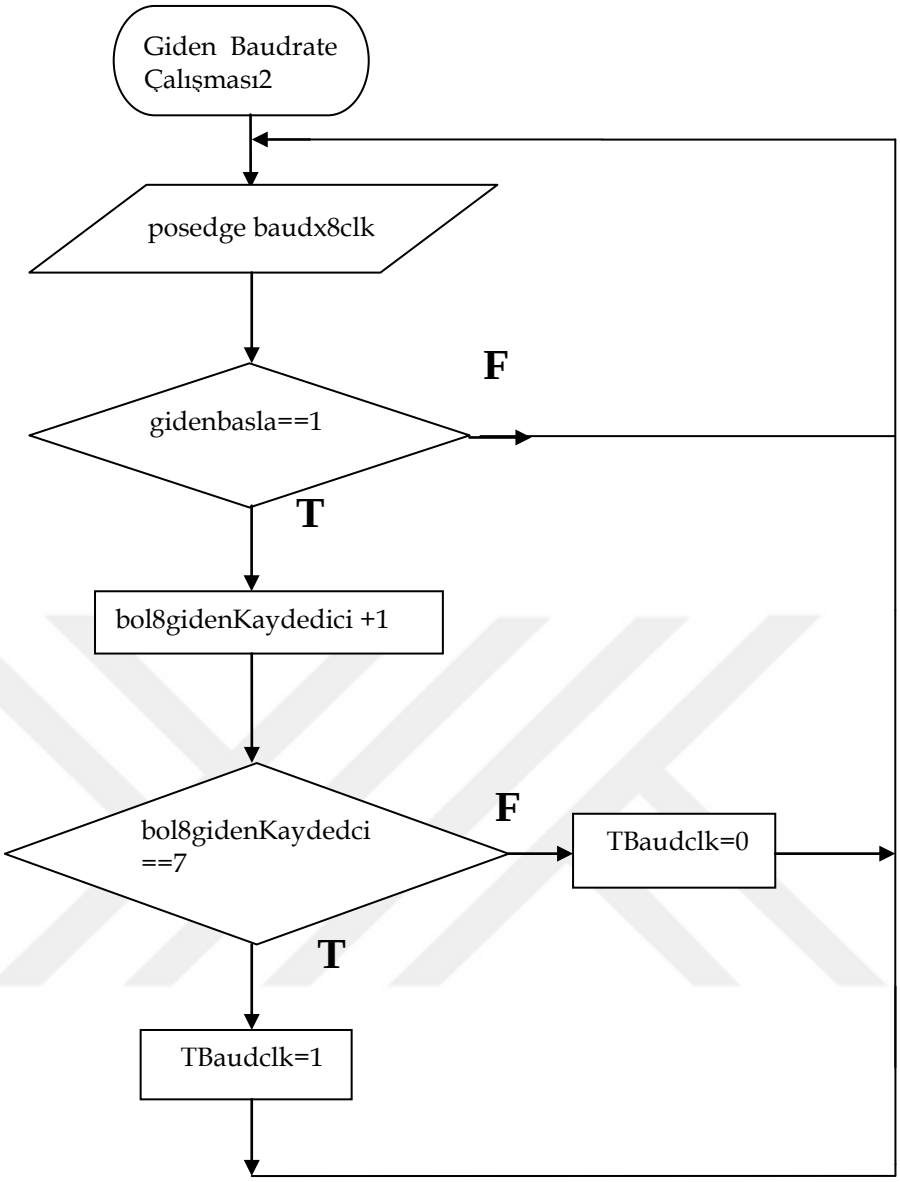


Şekil 3.12: Giden baudrate üretecinin algoritması.

Baudrate birim zaman içinde yollanan yada alınan bit sayısını belirleyen bir haberleşme hızı birimidir. 9600 baudrate denildiğinde bir saniye içinde 9600 tane bit gönderilmiş yada alınmış anlamına gelmektedir [23]. Bu tanımdan da anlaşıldığı gibi gelen ve giden baudrate üretecinin çalışma mantığı ve frekansı aynıdır. Eğer gelen ve giden baudrate üreteci farklı veri iletim frekansları için yapılandırılmışsa UART iletişimi çalışmaz.

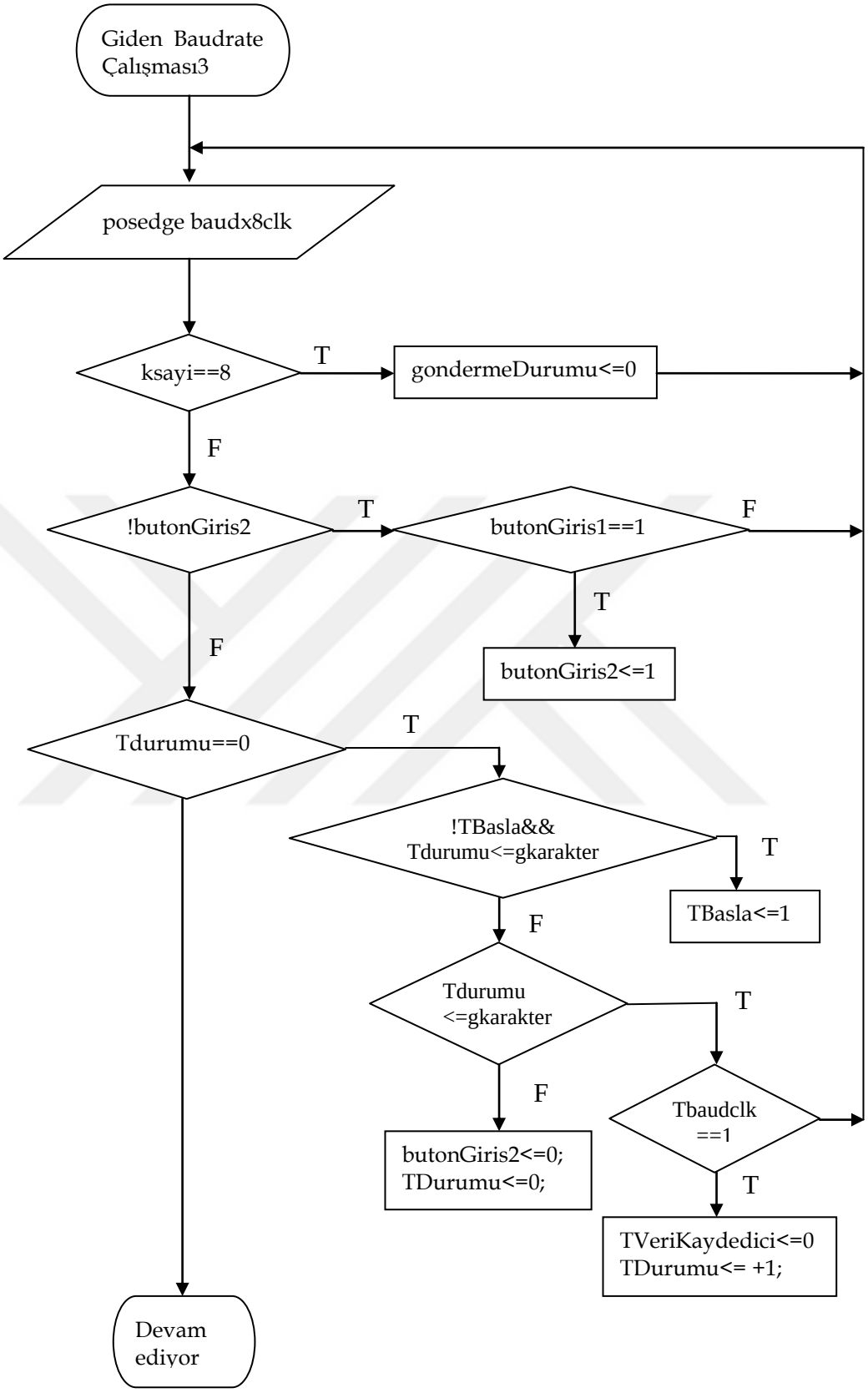


Şekil 3.13: UART modulü.

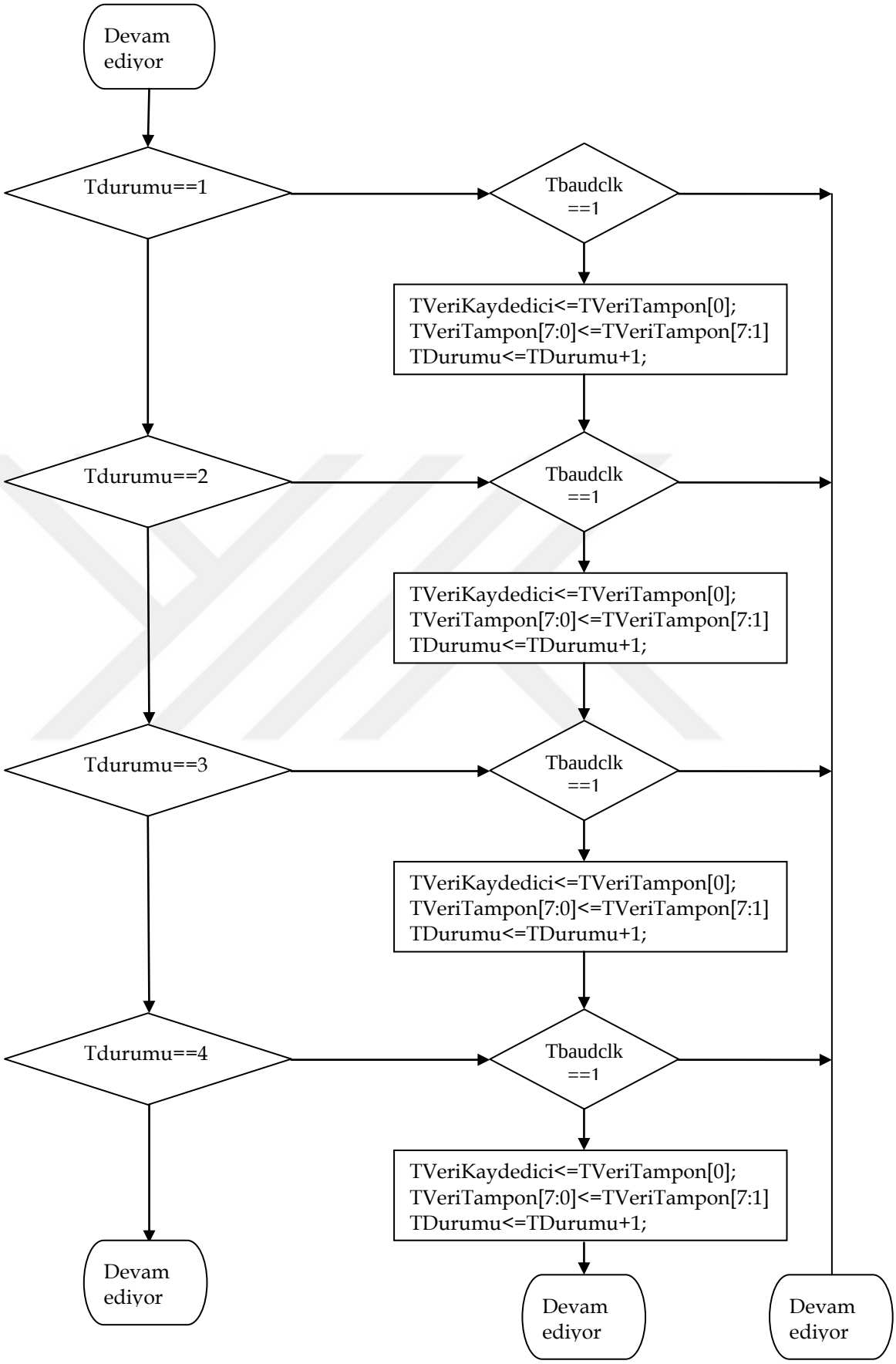


Şekil 3.14: Giden2 baudrate üretecinin algoritması.

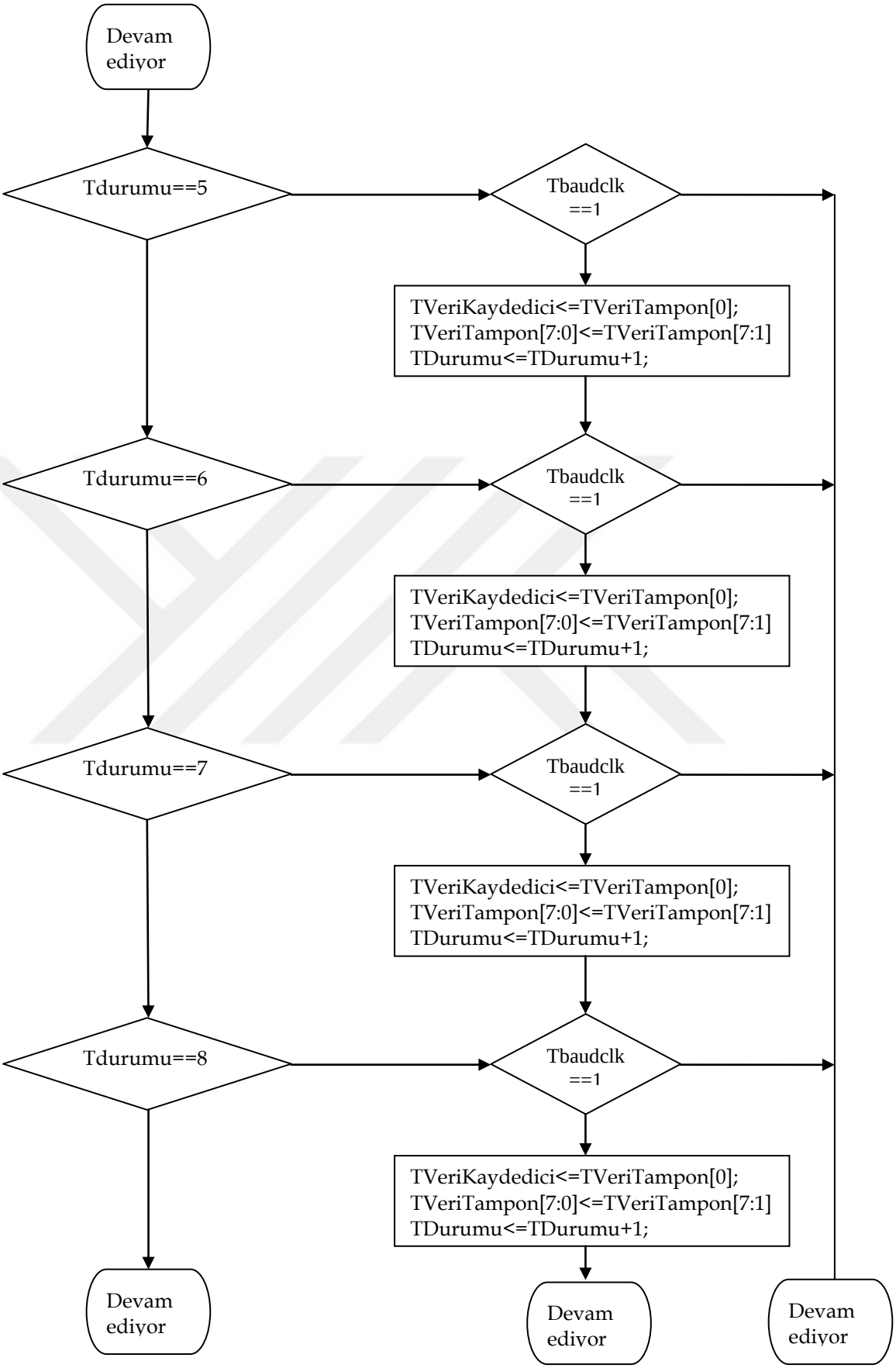
Yukarıdaki algoritma oluşan her baudx8clk sinyalinin yükselen kenarında programa girer. Programda ne zaman bilgi gönderilecekse gidenbasla değişkeni 1 yapılır. gidenbasla değişkeni 0 ise program hiçbir şey yapmadan döngünün başına döner. gidenbasla değişkeni 1 ise her 650 tane yükselen kenarda bol8gidenKaydedici değişkeni 1 artar. bol8gidenKaydedici değişkeni 7'ye eşit ise TBaudclk değişkeni 1'e eşitlenir. Aksi halde 0'a eşitlenir. TBaudclk değişkenin 1'e eşit olması bir bitlik sürenin oluştuğu anlamına gelir. 7'e kadar sayınca bir 2*325 kadar kare dalga gelince toplam 5205 tane peryot oluşmuş olur.



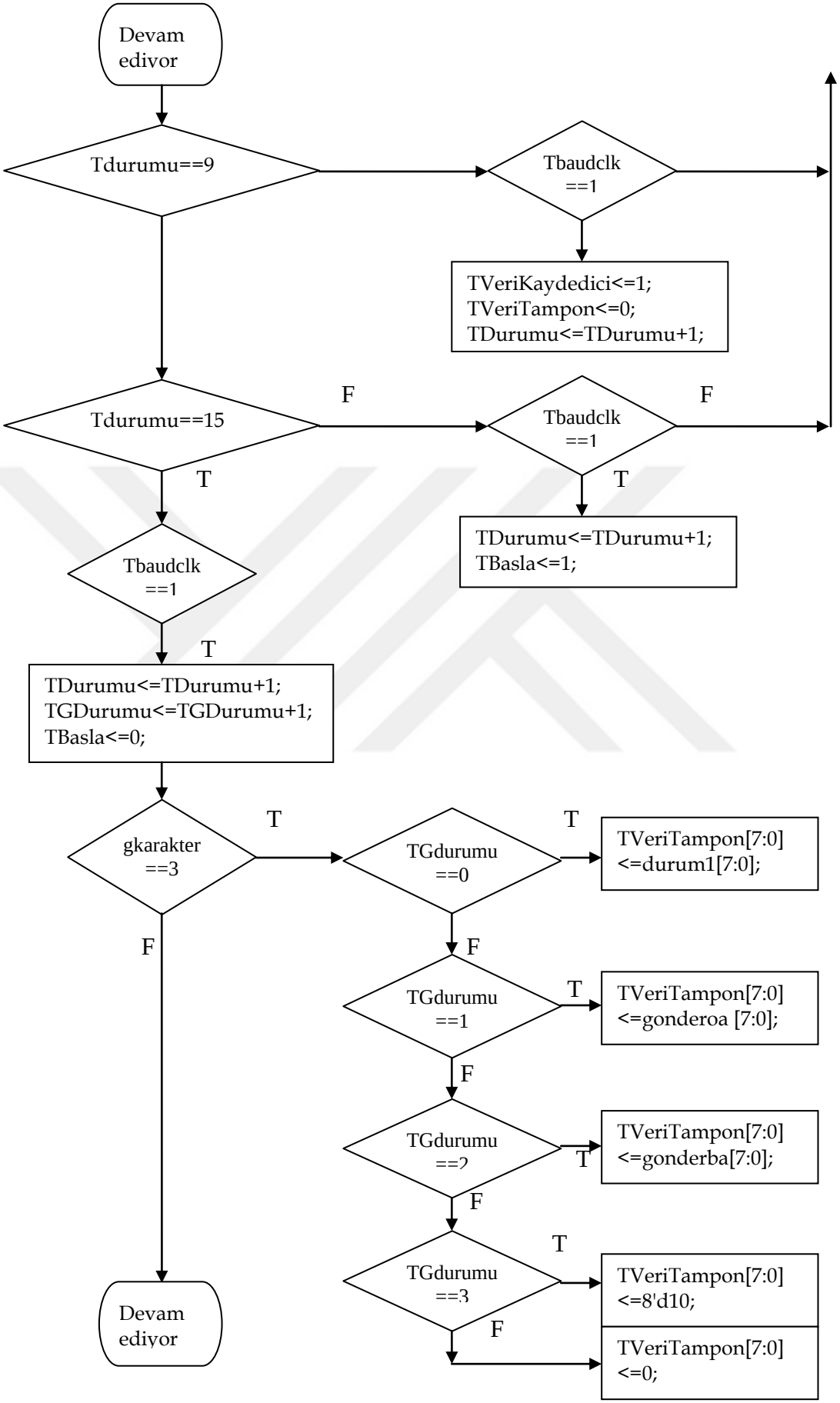
Şekil 3.15: Bilgisayardan giden bilgi algoritması.



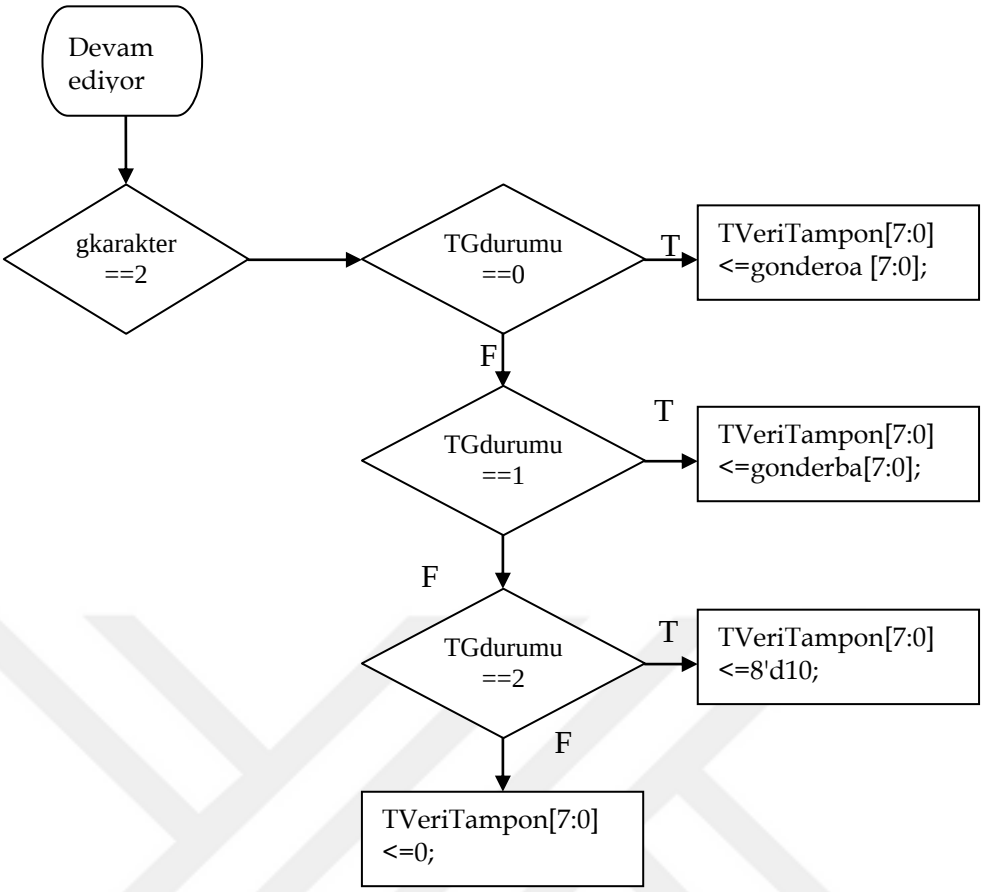
Şekil 3.15: Devam.



Şekil 3.15: Devam.



Şekil 3.15: Devam.



Şekil 3.15: Devam.

Programın bu kısmı 5205 kare dalgada bir kere döngüye girer. ksayi==8 durumu kontrol edilerek program bilgi göndermeye yeniden kurulur. Programda bilgi göndermek için başlangıç ve sonuç bayraklarına ihtiyaç vardır. Aksi taktirde aynı bilgi karşı tarafa birçok defa gereksiz şekilde gönderilecektir. ksayi==8 durumu bilgi gönderilmiş ise TDurumu değişkeni azami değeri almıştır. Yeni gönderilecek bilgiler için TDurumu değişkeni sıfırlanması gerekir.

Programa bilgi göndermeye başlaması için irisButon değişkeninin set edilmesi gerekir. irisButon değişkeni set edildiğinde butonGiris1 değişkeni de setlenir. Bu durumda başlangıçta sıfıra eşit olan butonGiris2 değişkeni 1'e eşitlenir. Böylece program !butonGiris2 durumunu sağlamadığı için sonraki şart koşuluna gider.

Başlangıçta TDurumu sıfıra eşit olduğu ve ilk bayt için baudrate üreteci Tbasla setlenir. Program giden bilgiler için kullanılan çıkış pinini 5205 kare dalga süresince başlama biti olan sıfıra çeker ve TDurumu bir arttırır.

5205 kare dalga sonra TDurumu==1 durumunda gönderilecek bilginin 0. biti çıkış pinine atanır (TVeriKaydedici<=TVeriTampon[0]). Aynı anda gönderilecek

bilginin bitleri bir bit sağa kaydırılır (TVeriTampon[7:0]<=TVeriTampon[7:1]). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

5205 kare dalga sonra TDurumu==2 durumunda gönderilecek bilginin 1. biti çıkış pinine atanır (TVeriKaydedici<=TVeriTampon[0]). Aynı anda gönderilecek bilginin bitleri bir bit sağa kaydırılır (TVeriTampon[7:0]<=TVeriTampon[7:1]). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

5205 kare dalga sonra TDurumu==3 durumunda gönderilecek bilginin 2. biti çıkış pinine atanır (TVeriKaydedici<=TVeriTampon[0]). Aynı anda gönderilecek bilginin bitleri bir bit sağa kaydırılır (TVeriTampon[7:0]<=TVeriTampon[7:1]). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

5205 kare dalga sonra TDurumu==4 durumunda gönderilecek bilginin 3. biti çıkış pinine atanır (TVeriKaydedici<=TVeriTampon[0]). Aynı anda gönderilecek bilginin bitleri bir bit sağa kaydırılır (TVeriTampon[7:0]<=TVeriTampon[7:1]). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

5205 kare dalga sonra TDurumu==5 durumunda gönderilecek bilginin 4. biti çıkış pinine atanır (TVeriKaydedici<=TVeriTampon[0]). Aynı anda gönderilecek bilginin bitleri bir bit sağa kaydırılır (TVeriTampon[7:0]<=TVeriTampon[7:1]). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

5205 kare dalga sonra TDurumu==6 durumunda gönderilecek bilginin 5. biti çıkış pinine atanır (TVeriKaydedici<=TVeriTampon[0]). Aynı anda gönderilecek bilginin bitleri bir bit sağa kaydırılır (TVeriTampon[7:0]<=TVeriTampon[7:1]). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

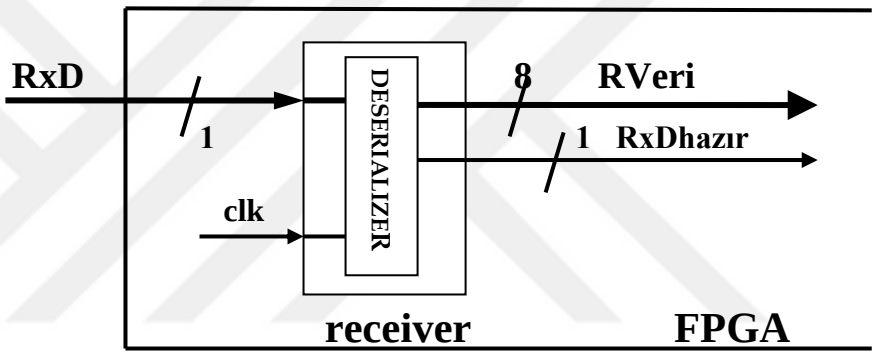
5205 kare dalga sonra TDurumu==7 durumunda gönderilecek bilginin 6. biti çıkış pinine atanır (TVeriKaydedici<=TVeriTampon[0]). Aynı anda gönderilecek bilginin bitleri bir bit sağa kaydırılır (TVeriTampon[7:0]<=TVeriTampon[7:1]). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

5205 kare dalga sonra TDurumu==8 durumunda gönderilecek bilginin en son biti olan 7. biti çıkış pinine atanır (TVeriKaydedici<=TVeriTampon[0]). Aynı anda gönderilecek bilginin bitleri bir bit sağa kaydırılır (TVeriTampon[7:0]<=TVeriTampon[7:1]). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

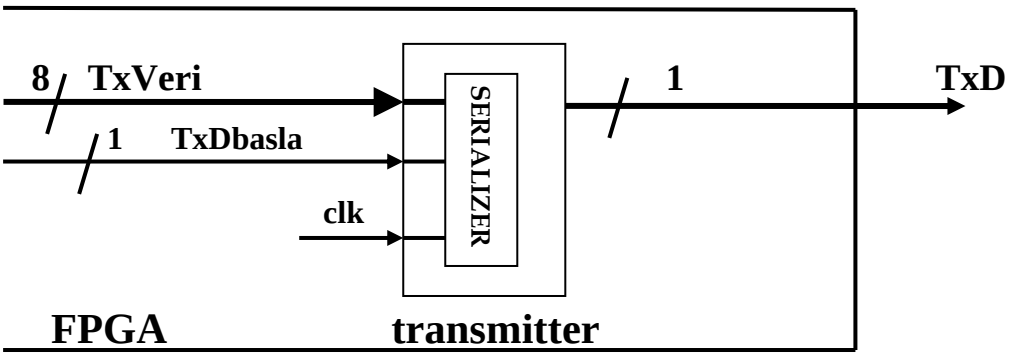
5205 kare dalga sonra TDurumu==9 durumunda gönderilecek bilginin bütün bitleri gönderildiği için hat 1'e çekilir (TVeriKaydedici<=1). TDurumu bir arttırılır (TDurumu<=TDurumu+1).

Tdurumu==15 olana kadar çıkış pinine bir değer atanmaz. Tdurumu==15 de Tdurumu bir arttırılarak dört bitlik 15 değerini geçemeyeceğinden dolayı sıfırlanır. Gkarakter==2 ise önce onlar basamağının ASCII kodu, ikinci olarak birler basamağının ASCII kodu ve son olarak karşı tarafa gönderilecek rakamların son bulduğu anlamına gelen yeni satır karakterinin (\n) ASCII kodu gönderilir.

Gkarakter==3 ise önce yüzler basamağının ASCII kodu, ikinci olarak onlar basamağının ASCII kodu üçüncü olarak birler basamağının ASCII kodu ve son olarak karşı tarafa gönderilecek rakamların son bulduğu anlamına gelen yeni satır karakterinin (\n) ASCII kodu gönderilir. İhtiyaten TGdurumu gereken durumları almazsa boş karakter gönderilir.



Şekil 3. 16: Receiver blok diyagramı.



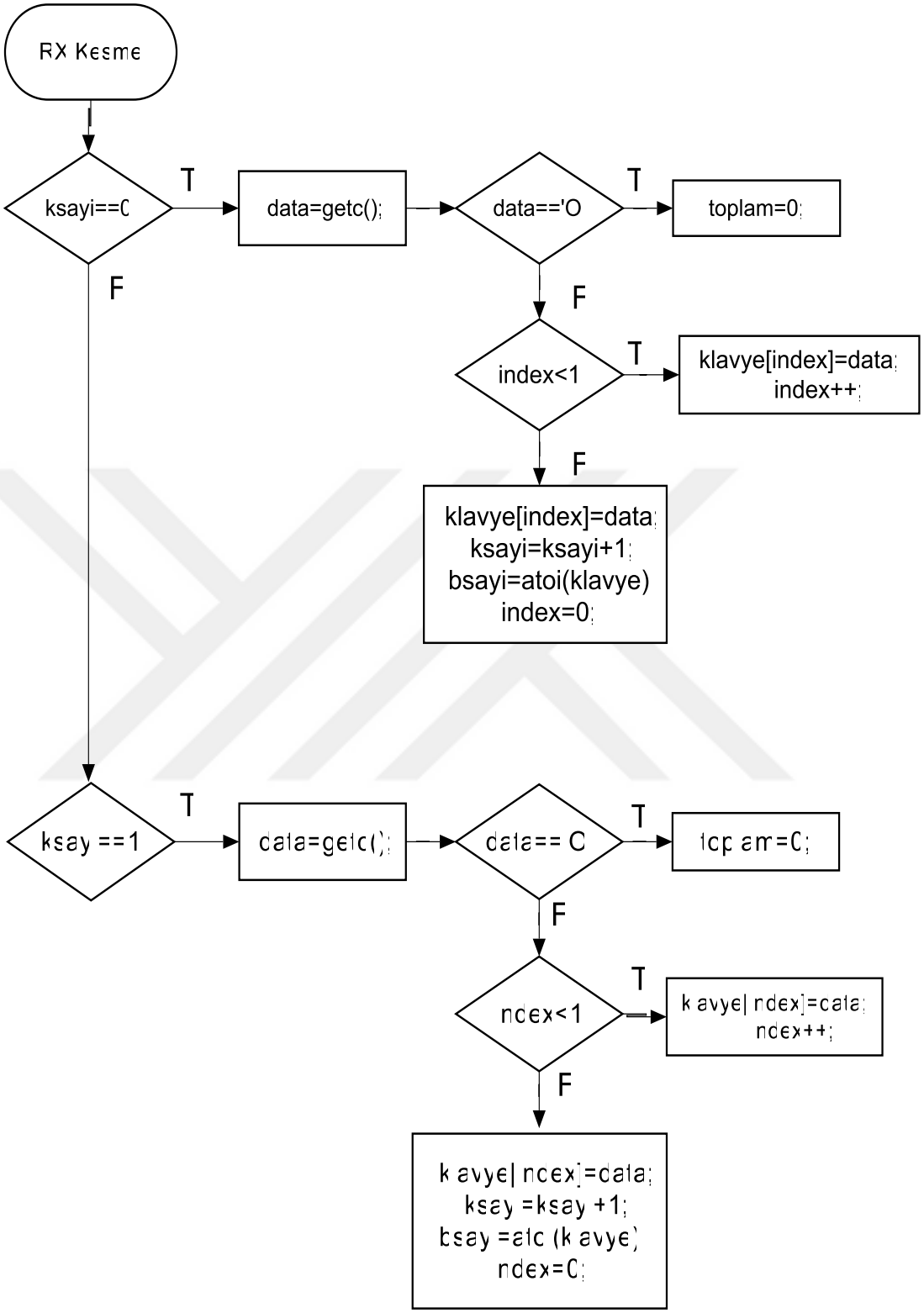
Şekil 3. 17: Transmitter blok diyagramı.

3.2. Mikrodenetleyicinin Seri Haberleşmesi

Seri haberleşme için mikrodenetleyici kullanırken PIC ailesini derlemek için C dilini kullanan CCS C derleyicisinin hazır kütüphaneleri kullanıldı. CCS C'de seri haberleşme kesmesi kullanıldı. Mikrodenetleyicinin seri veri gelen pinine bilgi geldiğinde program bu seri haberleşme kesmesi içine girmektedir. Seri haberleşme kesmesi içinde ksayi==0 ise birinci sayı, ksayi==1 ise ikinci sayı olup olmadığı tespit edilir. ksayi==0 durumunda ilk olarak gelen veri sıfıra eşit ise bilgisayar arayüzünden resetleme butonuna basıldığı anlaşılır ve programdaki değişkenler sıfırlanır. Mikrodenetletici gelen bilgiler karakter karakter alınarak index<1 durumunda birler basamağı aksi durumda onlar basmağı alınarak klavye adlı değişkenin sıfırıncı ve birinci indekslerine yerleştirir. Onlar basamağı yerleştirildikten sonra matematiksel işlem yapılabilmesi için atoi fonksiyonuyla string olan klavye dizisi integer türüne dönüştürüldü [22].

ksayi==1 durumunda ilk olarak gelen veri sıfıra eşit ise bilgisayar arayüzünden resetleme butonuna basıldığı anlaşılır ve programdaki değişkenler sıfırlanır. Mikrodenetletici gelen bilgiler karakter karakter alınarak index<1 durumunda birler basamağı aksi durumda onlar basmağı alınarak klavye adlı değişkenin sıfırıncı ve birinci indekslerine yerleştirir. Onlar basamağı yerleştirildikten sonra matematiksel işlem yapılabilmesi için atoi fonksiyonuyla string olan klavye dizisi integer türüne dönüştürülür.

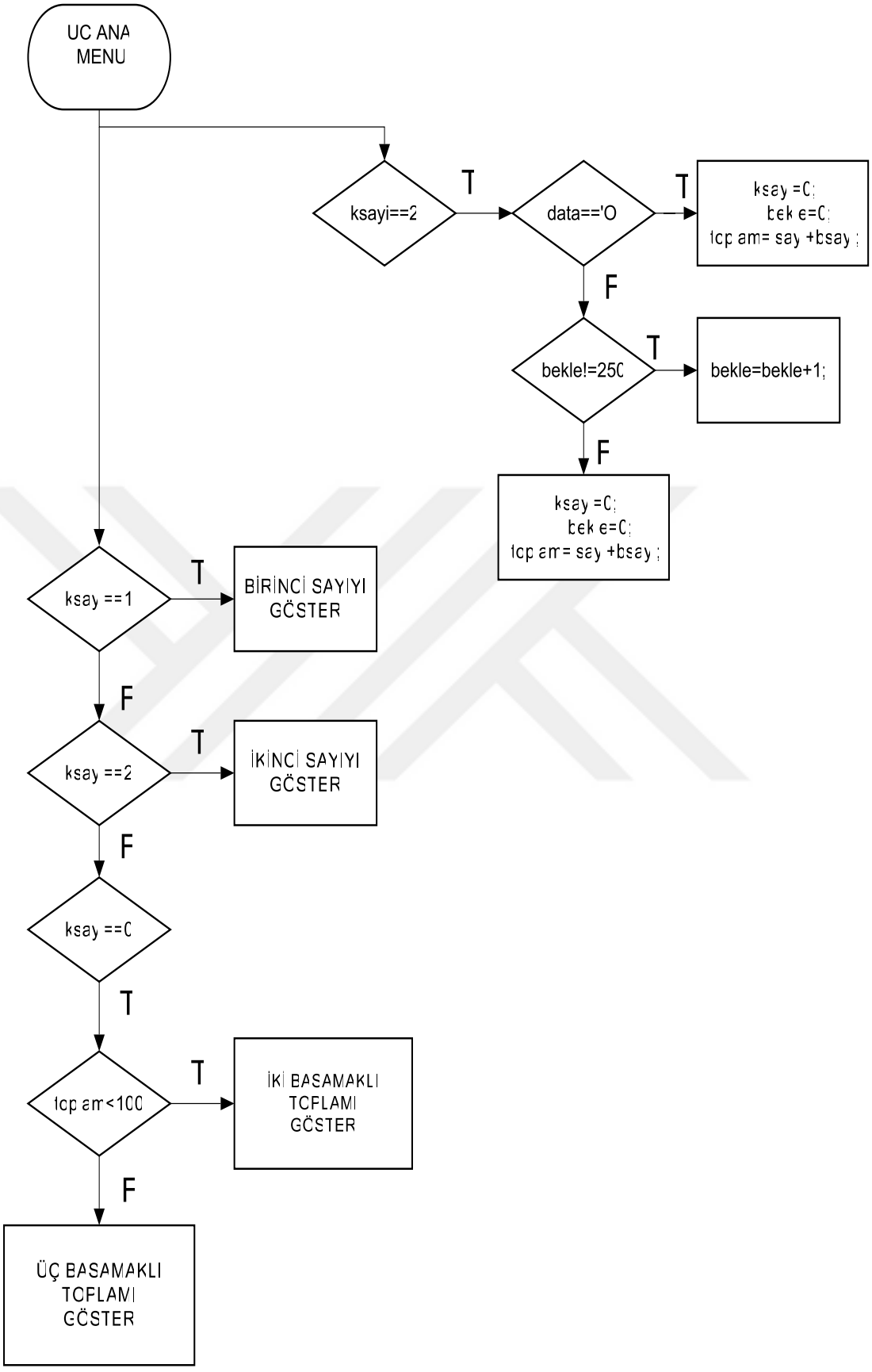
Programın ana kısmında ksayi==2 durumunda gelen bilginin sıfıra eşit olması sorgulanır. Bu durumda değişkenler kesmede sıfırlandığı için toplam sonuç hemen 7s displayde gösterilir. Aksi durumda ikinci sayı dispayde bir süre gözükmesi için küçük bir gecikme programı yazılır. ksayi==1 durumunda yani ilk iki basmaklı sayı geldiğinde 10'a bölünerek onlar basmağı, mod10 alınarak birler basmağı bulunur. Birler basamağının gösterimi için ortak katotlu displayin ortak ucu setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. 5 ms sonra birler basamağının gösterimi için ortak katotlu displayin ortak ucu resetlenir, onlar basamağının gösterimi için ortak katotlu displayin ortak ucu ise setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. Yeni sayı gelene kadar bu şekilde sürekli program kendini tekrar eder.



Şekil 3.18: Seri haberleşme kesme algoritması.

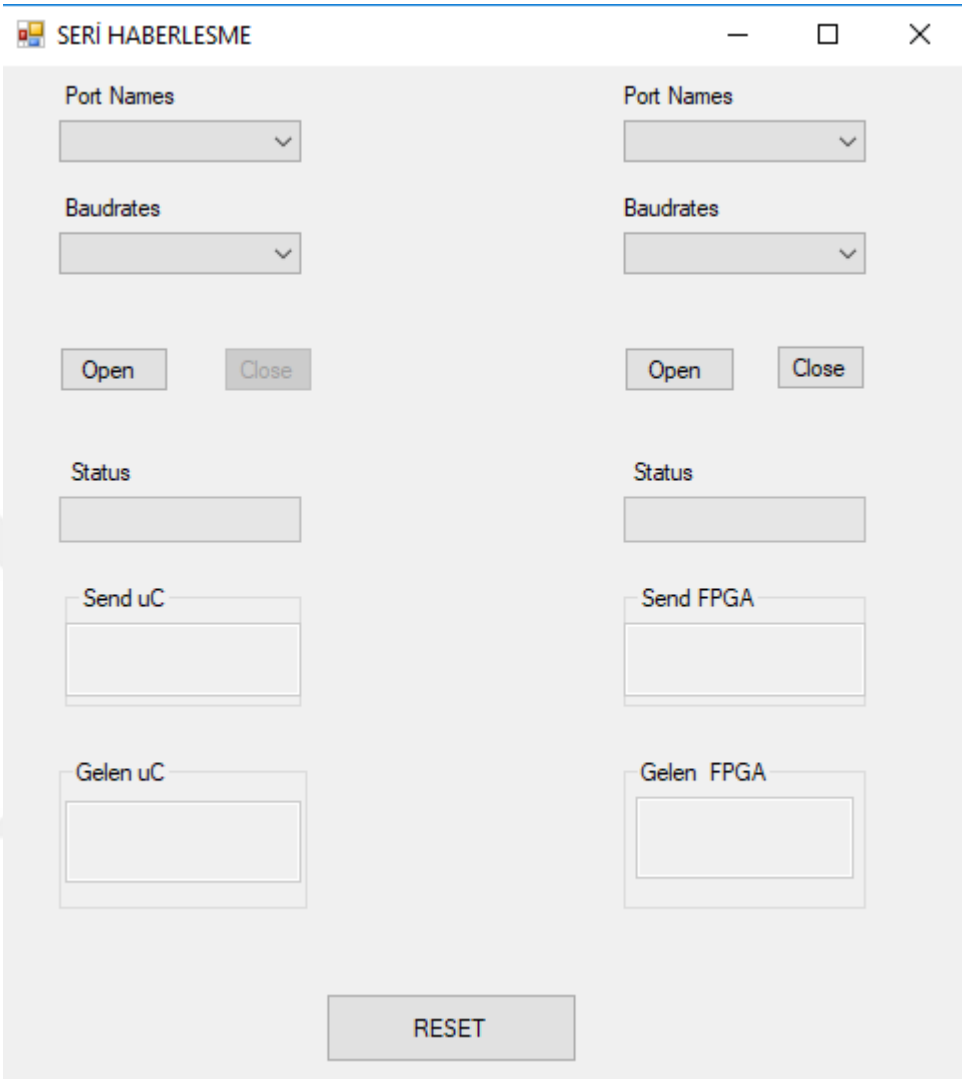
ksayi==2 durumunda yani ikinci iki basmaklı sayı geldiğinde 10'a bölünerek onlar basmağı, mod10 alınarak birler basmağı bulunur. Birler basamağının gösterimi için ortak katotlu displayin ortak ucu setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. 5 ms sonra birler basamağının gösterimi için ortak katotlu displayin ortak ucu resetlenir, onlar basamağının gösterimi için ortak katotlu displayin ortak ucu ise setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. Yeni sayı gelene kadar bu şekilde sürekli program kendini tekrar eder.

ksayi==0 durumunda yani mikrodenetleticiye iki sayı geldiğinde toplanır. Toplam eğer iki basmaklı sayı ise toplam 10'a bölünerek onlar basmağı, mod10 alınarak birler basmağı bulunur. Birler basamağının gösterimi için ortak katotlu displayin ortak ucu setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. 5 ms sonra birler basamağının gösterimi için ortak katotlu displayin ortak ucu resetlenir, onlar basamağının gösterimi için ortak katotlu displayin ortak ucu ise setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. Yeni sayı gelene kadar bu şekilde sürekli program kendini tekrar eder. Toplam eğer üç basmaklı sayı ise toplamdan 100 çıkarıp 10'a bölünerek onlar basmağı, mod10 alınarak birler basmağı bulunur. Birler basamağının gösterimi için ortak katotlu displayin ortak ucu setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. 5 ms sonra birler basamağının gösterimi için ortak katotlu displayin ortak ucu resetlenir, onlar basamağının gösterimi için ortak katotlu displayin ortak ucu ise setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. 5 ms sonra onlar basamağının gösterimi için ortak katotlu displayin ortak ucu resetlenir, yüzler basamağının gösterimi için ortak katotlu displayin ortak ucu ise setlenir. Programın başında decimal kodların 7s displayde gözükmesi için gerekli hex kodları çıkış portlarına gönderilir. Yeni sayı gelene kadar bu şekilde sürekli program kendini tekrar eder.



Şekil 3.19: Mikrodenetleyici çalışma algoritması.

3.3. Seri Haberleşme Bilgisayar Arayüzü Çalışma Prensibi



Şekil 3.20: Seri haberleşme bilgisayar arayüzü.

C# ’da seri haberleşme için gerekli olan giriş çıkış pinlerini kullanabilmek için programın başına using System.IO.Ports kütüphanesini eklendi. GetPortNames fonksiyonu mevcut sistemdeki tüm seri port adlarını string türünde veren fonksiyondur [19]. Bu fonksiyon ile elde edilen bilgiler comboboxlarda gösterilir.

Open butonuna basıldığında port names adlı comboboxdan mikrodenetleyicinin bağlandığı kontrol edilir. Mikrodenetletici ile bilgisayar arasındaki uart haberleşmesi için gerekli baudrate oranı kontrol edilir. Port names ve baudrates seçilmemişse uyarı mesajı verilir. Eğer her iki tercih yapılmışsa combobox1’den gerekli usb portu serial port1’e atanır. Baudrates string bir ifade

olduğu için int32 türüne dönüştürülür. Serial port1 veri alış-verişine açılır. Status durum çubuğu yüzde yüz aktif edilir ve open butonu pasif, close butonu ve send uc textbox'ı aktif edilir.

Close butonuna basıldığında Serial port1 veri alış-verişine kapanır. Status durum çubuğu yüzde yüz pasif edilir ve open butonu aktif, close butonu ve send uc textbox'ı pasif edilir.

Send uc textbox1'ne girilen sayılar entera basılarak seri port1'e gönderilir. Textbox1 içeriği temizlenir ve çıkış portu bufferı resetlenir.

Send FPGA textbox1'ne girilen sayılar entera basılarak seri port2'ye gönderilir. Textbox3 içeriği temizlenir ve çıkış portu bufferı resetlenir.

Seri port1 girişine bilgi geldiğinde SerialDataReceivedEventArgs olayı gerçekleşir. Program bu olayın olduğunda gerçekleşecek komut satırını işler. Burada readline komutu yardımıyla seri port1'e gelen bilgi newline karakteri gelene kadar okunur ve okunan değer sb değişkenine atanır. Okunan bu değer Gelen uc textbox içerisinde gösterilir. Si değişkeni sb değişkenine eşit değilse FPGA aygıtının bağlı bulunduğu seri port2'ye önce ayrı ayrı iki tane 0 karakteri daha sonra ise sb değişkeni yazdırılır. Bu şekilde mikrodenetleyici içerisindeki bilgi bilgisayar ara yüzü kullanarak FPGA gönderilir.

Seri port2 girişine bilgi geldiğinde SerialDataReceivedEventArgs olayı gerçekleşir. Program bu olayın olduğunda gerçekleşecek komut satırını işler. Burada readline komutu yardımıyla seri port2'e yani FPGA aygıtına gelen bilgi newline karakteri gelene kadar okunur ve okunan değer si değişkenine atanır. Okunan bu değer Gelen FPGA textbox içerisinde gösterilir. Si değişkeni sb değişkenine eşit değilse mikrodenetleyicinin bağlı bulunduğu seri port1'ye önce iki tane 0 karakteri daha sonra ise si değişkeni yazdırılır. Bu şekilde FPGA içerisindeki bilgi bilgisayar ara yüzü kullanarak mikrodenetleyiciye gönderilir.

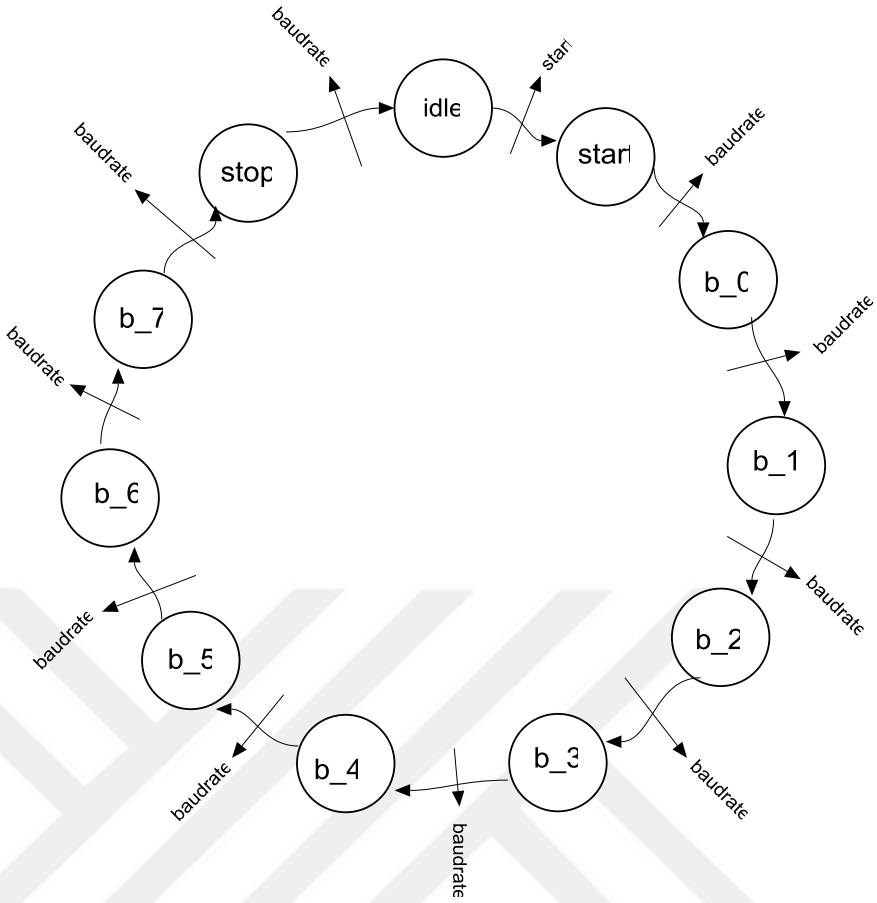
Reset butonuna basıldığında mikrodenetleyicinin bağlı olduğu seri port1'e 0 karakteri, FPGA aygıtının bağlı olduğu seri port2'e ise ayrı ayrı dört tane 0 karakteri gönderilir. Böylece her iki elemanın içeriği resetlenmiş olur.

4. MİKRODENETLEYİCİ – BİLGİSAYAR - FPGA SERİ HABERLEŞMESİNE ÖRNEK UYGULAMA

Tezin bu kısmında önceden oluşturduğumuz UART çalışmasına bir örnek uygulama anlatılacaktır. Bu çalışmada mikrodenetleyici yardımıyla LM35 sıcaklık sensöründen ortamın sıcaklığı ölçüldü. Ölçülen değer mikrodenetleyicinin seri haberleşme protokolleri kullanılarak bilgisayara gönderildi. C# dilinde yazılan arayüz mikrodenetleyicinin gönderdiği değeri ekranda göstermektedir. Ayrıca bu değeri bilgisayarın usb-uart çıkışına bağlı FPGA aygıtına gönderir. FPGA bu gelen değerlerin aritmetik ortalamasını alarak sonucu bilgisayara tekrar gönderir. Bilgisayar bu değerleri arayüz programında gösterir.

4.1. FPGA Transmitter FSM

Bilgi işlemine başlanabilmesi için FPGA elemanına mikrodenetleyicinin ölçtüğü sıcaklık değerinin gelmesi gerekir. Bu değer geldikten sonra bu değerin işleme tabi tutulması için gerekli çok kısa bir süre sonra bilgi gönderme işlemi başlar. FPGA elemanından bilgisayara bilgi gönderen VERILOG programının FSM diyagramına göre bilgi yokken hat idle yani boşda 1 durumunda bekler. Bilgi gönderilmek istendiğinde hat 0 durumuna getirilir. Hat baudrate miktarınca 0 durumunda beklenir. Gönderilecek bilginin ilk karakterinin en değersiz bitinden en değerli bitine olucak şekilde başlayarak her gönderilen bit arasında baudrate miktarınca beklenerek bilgi gönderilmeye başlanır. Her bit gönderme işlemi sıfırıncı bit çıkış pinine aktarılır. Bunun için her bit gönderildikten sonra sıradaki bitler sağa doğru bir bit kaydırılır. Sekiz bit gönderildiğinde hat 1 durumuna çekilir ve baudrate miktarından sonra yeni bilgi gönderme işlemine hat hazır olur. Burada ölçülen sıcaklık önce onlar basamağı ardından birler basamağı ve en son newline komutunun ASCII karakteri gönderilir.



Şekil 4.1: FPGA transmitter FSM.

4.2. FPGA Receiver FSM

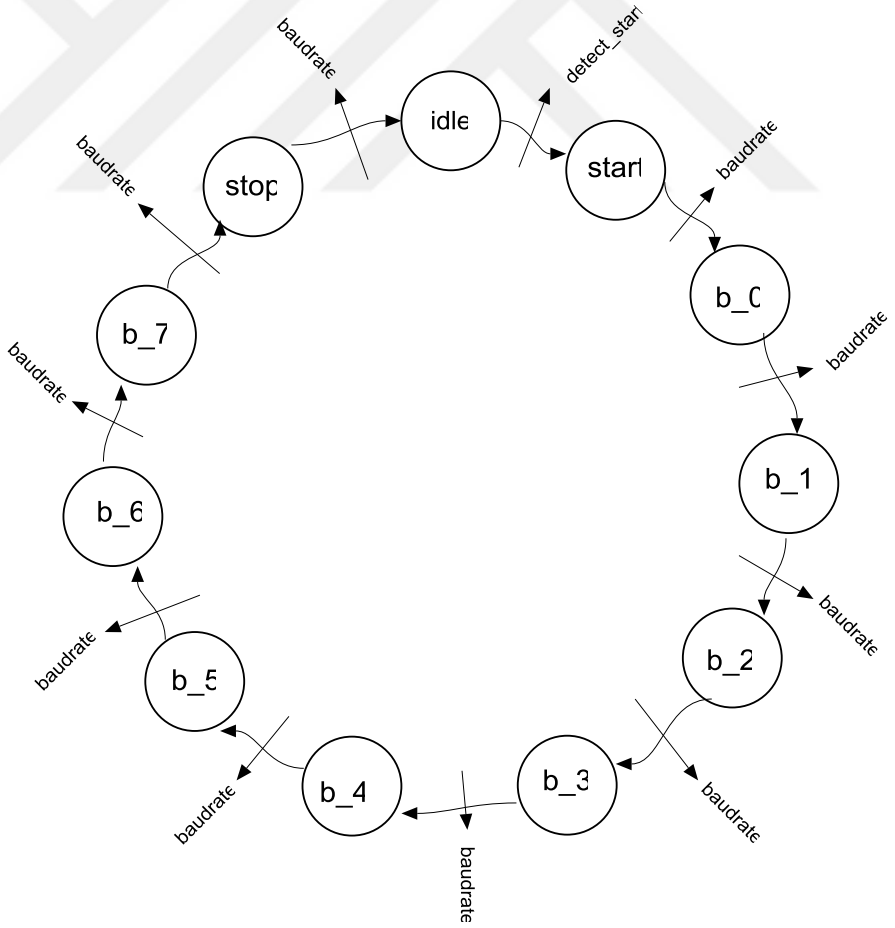
Karşı taraftan gönderilen sıcaklık bilgisinin alınabilmesi için FSM diyagramına göre bilgi gönderme işlemine başlandığının işaretinin gelmesi gerekir. Bu işaret hattın 1 durumundan 0 durumuna geçmesi ve 0 durumunda baudrate miktarınca beklemesidir. Bilgi gelme işaretinden sonra karşı taraftan gelen bir baytlık yani 8 bitlik bilginin en değersiz biti olan 0. biti tanımlanan 8 bitlik bir dizinin 7. bitine yerleştirilir. Bu yerleştirme işlemiyle birlikte aynı anda bu dizinin bitleri bir bit sağa kaydırılır. 1. ve 7. bit gelene kadar gelen bitlere de bu işlemler uygulanır. 7. bitten sonra hat stop durumuna yani baudrate miktarınca 1 durumuna geçtiği için bilgi alma işlemini bitirir.

İlk gelen 8 bit sıcaklık değerinin onlar basamağı, ikinci gelen ise birler basamağı olacak şekilde değişkenlere atanır. Bu değişkenler ASCII kod olduğu için

decimal koda çevrilir. Çevrilen decimal kodlardan ilkkiolan onlar basamağı 10 ile çarpma işlemine tutulup ikincisi olan birler basamağı ile toplanarak sıcaklık değeri bulunur.

Gelen sıcaklık değeri sıfırlamadan sonra ilk ise bu değer hem toplam değer hem de ortalama değer sayılır. Gelen sıcaklık adeti 7'den küçük ise gelen sıcaklık değerleri toplama işlemine tutulur. Toplam sonucu gelen sıcaklık adetine bölerek sıcaklık değerlerinin ortalaması bulunur. Gelen sıcaklık adeti 7 ise gelen sıcaklık değerleri toplama işlemine tutulur. Toplam sonucu gelen sıcaklık adetine bölerek sıcaklık değerlerinin ortalaması bulunur. Sistem gelen sıcaklık adeti 8 tanede bir toplam sonucunu sıfırlar.

Ortalama değer, mod10 alarak birler basamağı, 10'a bölerek onlar basmağı bulunur. Bu değerler 7segment displayde gösterilir. Ayrıca bilgisayara gönderilebilmesi için decimal koddan ASCII koda çevrilir.



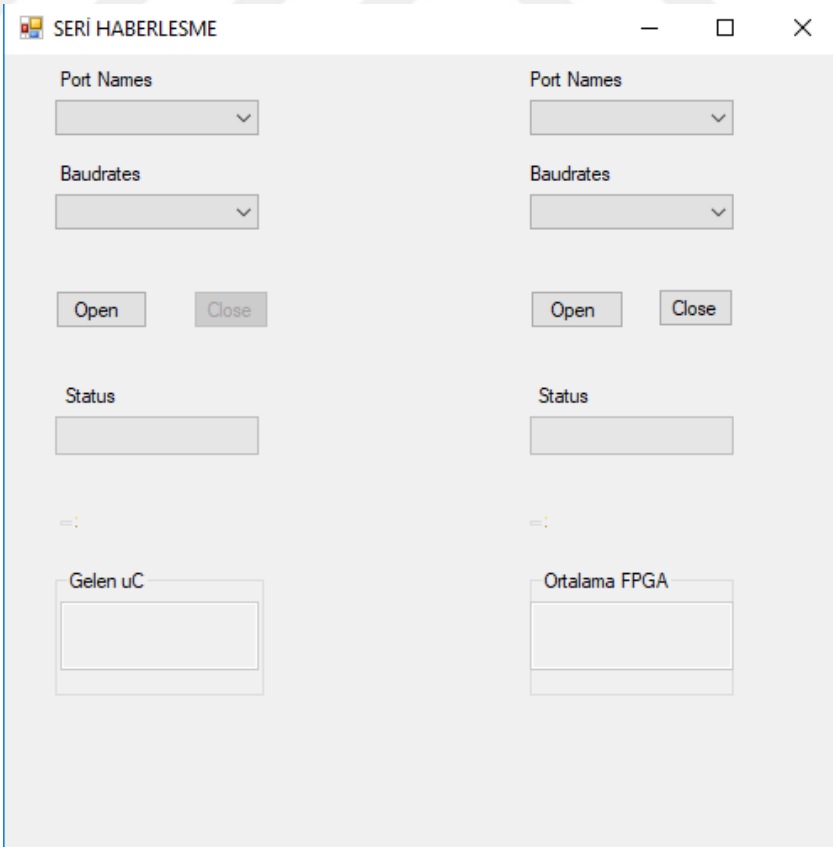
Şekil 4.2: FPGA receiver FSM.

4.3. Bilgisayar Grafik Arayüzü

Bilgisayar arayüz paneli C# programında yazıldı. Bilgisayarın port giriş çıkış işlemleri için gerekli sistem kütüphaneleri eklendi. Bilgisayarın usb girişlerine bağlanan aygıtlar panelde gözükmesi için gerekli program kodları yazıldı. Baudrates combobox'ından bilgisayarın diğer cihazlarla uart çalışma hızı seçildi. Open butonu ile seri port1 ve seri port2 iletişime açıldı ve status çubuğu aktif edildi. Close butonu ise seri port1ve seri port2 iletişime kapatır.

Seri port1'e bilgi geldiğinde bu bilgi sb değişkenine atanırken Gelen Uc adlı textbox'da da gösterilir. Gelen bilgi string olduğu için önce int ifadesine çevrilir. İnt türüne çevrildikten sonra birler ve onlar basamağı bulunur. Basamak çözümleme işleminden sonra birler ve onlar basamağı tekrar string türüne çevrilerek önce onlar basamağı sonra birler basamağı olmak üzere seri port2'ye gönderilir.

Seri port2'ye bilgi geldiğinde ise newline karakterine kadar okunur ve Ortalama FPGA textbox'ına yazılır.



Şekil 4.3: Bilgisayar grafik arayüzü.

4.4. Seri Haberleşme Noktası

Günümüzde üretilen bilgisayarlara genellikle seri port çıkışı bulunmamaktadır. Bunun yerine USB portları vardır. Bilgisayar ile FPGA ve mikrodenetleyici arasındaki bağlantıyı sağlamak için FT232R USB seri UART arayüzü kullanılır. Asenkron seri veriyi tek çiple USB transfer arayüzü sağlar. Bilgisayarın tanıması için bir program yüklemeye gerek yoktur. Mikrodenetleyici ve FPGA'e bağlıyken harici saat palsi gerektirmez. Board üzerinde transmit ve receive ledleri vardır. Bu ledlere gözlemlenerek haberleşme hakkında yorum yapılabilir. FT232RL USB UART arayüzü, FPGA için 3V3 fakat mikrodenetleyici için 5V gerilim değerinde çalışacak jumper seçimi yapılmalıdır. Aksi takdirde seri iletişim çalışmayacaktır. Çalışma hızı 300 baud ve 300 Mbaud hızına ulaşabilir. Tek, çift, no parity bit, 1 veya 2 stop bit, 7 veya 8 bit çalışmayı desteklemektedir.

Şekil 4.4: FT232RL USB UART dönüştürücü.

4.5. Mikrodenetleyicinin Çalışması

Çalışmalarda pratiklik sağlaması amacıyla sıcaklık ölçümünde Arduino kullanıldı. LM35 serisi, çıkış gerilimi santigrat sıcaklığına doğrusal olarak orantılı olan hassas entegre devre sıcaklık sensörleridir. Her 1 santigrat derece değişiminde 10 mV çıkış gerilimi değişikliği ile sıcaklığa doğrusal orantılı bir çıktı sağlar. LM35'lerin kullanımı, termistörlerden ve termokupllardan daha kolaydır, çünkü çok doğrusaldırlar ve sinyal işleme gerektirmezler.

Bir LM35'in çıkışı doğrudan bir Arduino analog girişine bağlanabilir. Arduino analog-dijital dönüştürücü (ADC) 1024 bit çözünürlüğe sahiptir.

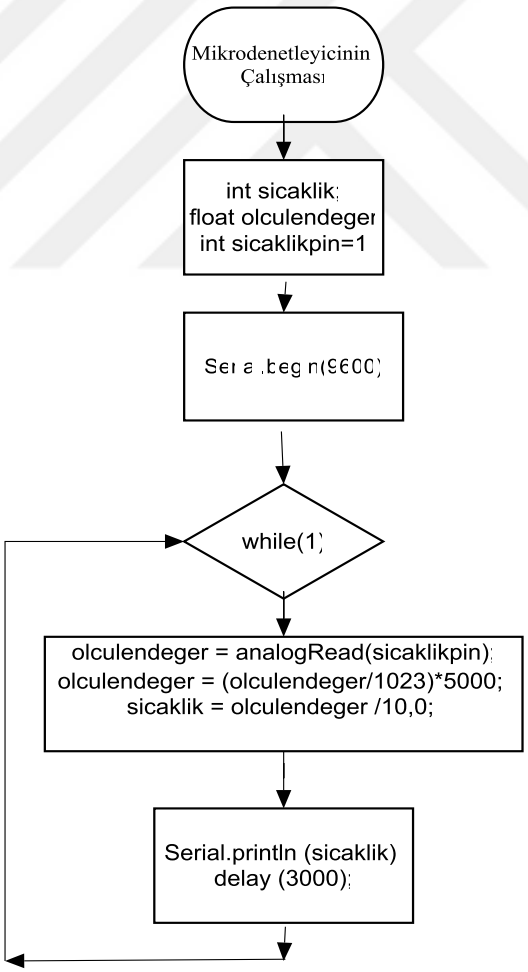
Buradan;

5/1024 = 0.00488

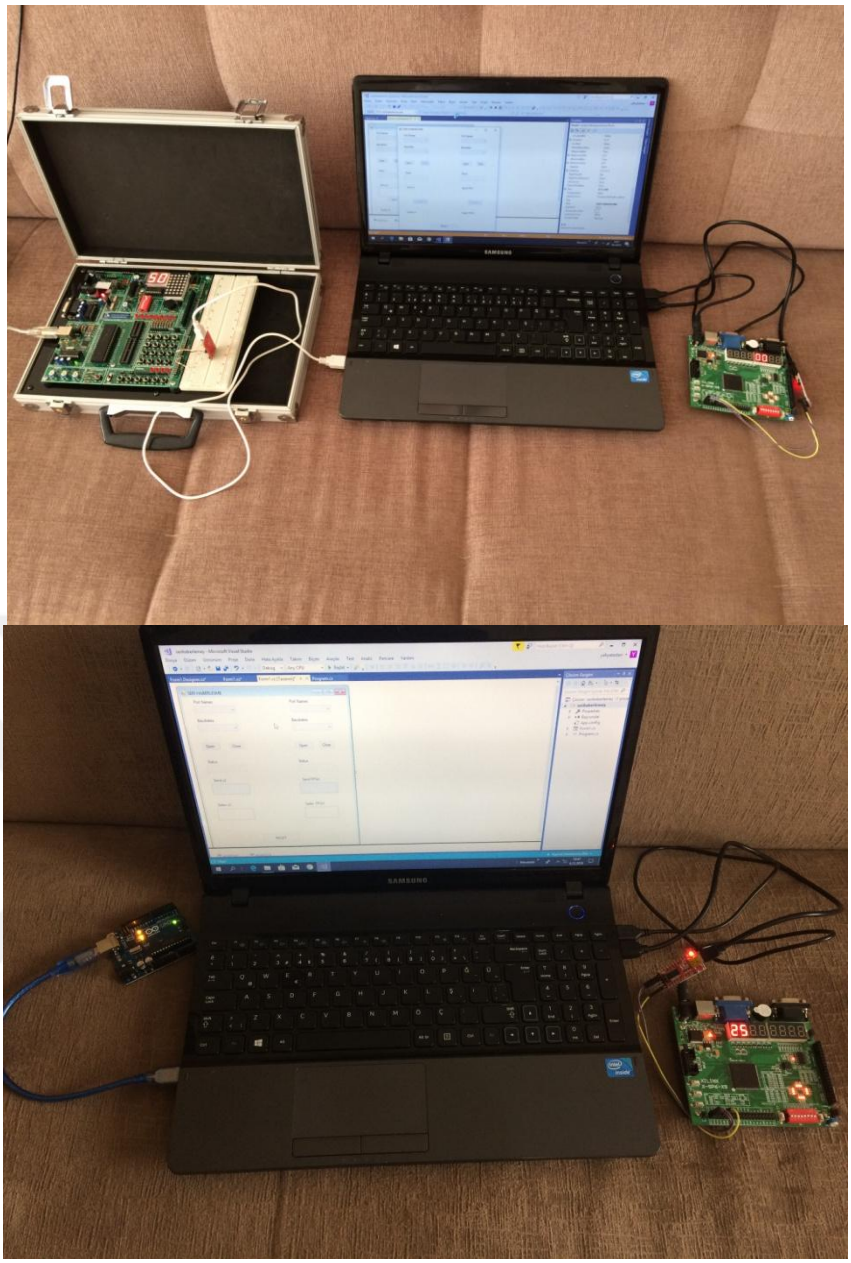
0.488 mV her bir sayısal analog veriye karşılık gelmektedir. Analog olarak okuduğunan değeri 0.488 mV ile çarpılırsa sonuç LM35'den okunan mV değerini verecektir. Şimdi sıcaklık değeri için bir işlem daha yapılması gerekir. LM35, her bir santigrat için 10mV değer üretmektedir. Bunun için ölçülen mV değerini 10'a bölünürse sonuç olarak santigrat cinsinden sıcaklık değerini bulmuş olunur.

santigrat derece = (analog bilgi x 0.488) / 10

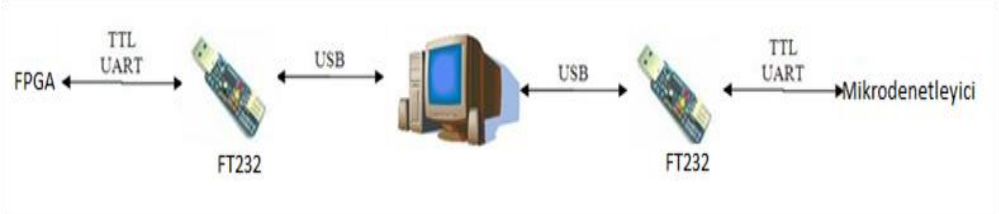
Programın girişinde analog değeri dönüştürülecek sıcaklık değerini tutan int sicaklik değişkeni, ölçülecek analog değeri tutan float olculendeger değişkeni ve analog değer girişi sicaklikpin=1 tanımlamaları yapıldı. Seri haberleşme hızı 9600 baudrate ayarlandı. Üç saniyede birkez Lm35 sensörü çıkış ucundan ölçülen analog değer mV değerine çevrildikten sonra bu değer de sıcaklık değerine dönüştürülerek seri port çıkışına gönderilir.



Şekil 4.5: Mikrodenetleyicinin çalışma algoritması.



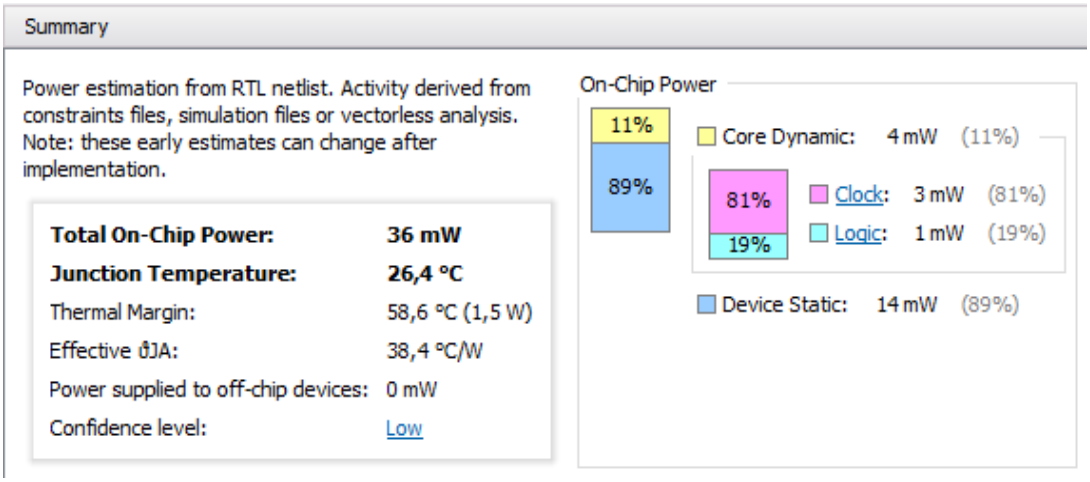
Şekil 4. 6: Çalışan sistemden görüntüler.



Şekil 4. 7: Sistemin blok diyagramı.

Tablo 4. 1: Advanced HDL Synthesis Report.

Macro Statistics		
# RAMs	16x8-bit single-port distributed Read Only RAM	5
# MACs	8x4-to-8-bit Dynamic Mult/MultAdd	1
# Adders/Subtractors	4-bit adder	2
	4-bit adder carry in	2
	8-bit adder carry in	32
	9-bit adder	1
	9-bit adder carry in	9
# Counters	16-bit up counter	1
	20-bit up counter	2
	3-bit up counter	3
	5-bit up counter	1
	6-bit up counter	1
# Accumulators	9-bit up accumulator	1
# Registers		103
Flip-Flops		
# Comparators	10-bit comparator lessequal	5
	11-bit comparator lessequal	5
	12-bit comparator lessequal	5
	20-bit comparator lessequal	1
	3-bit comparator greater	2
	4-bit comparator lessequal	2
	5-bit comparator greater	1
	8-bit comparator lessequal	20
	9-bit comparator lessequal	11
	1-bit 2-to-1 multiplexer	278
# Multiplexers	8-bit 2-to-1 multiplexer	16
	8-bit 4-to-1 multiplexer	1
	9-bit 2-to-1 multiplexer	3
Minimum period: 13.468ns (Maximum Frequency: 74.250MHz)		
Minimum input arrival time before clock: 4.742ns		



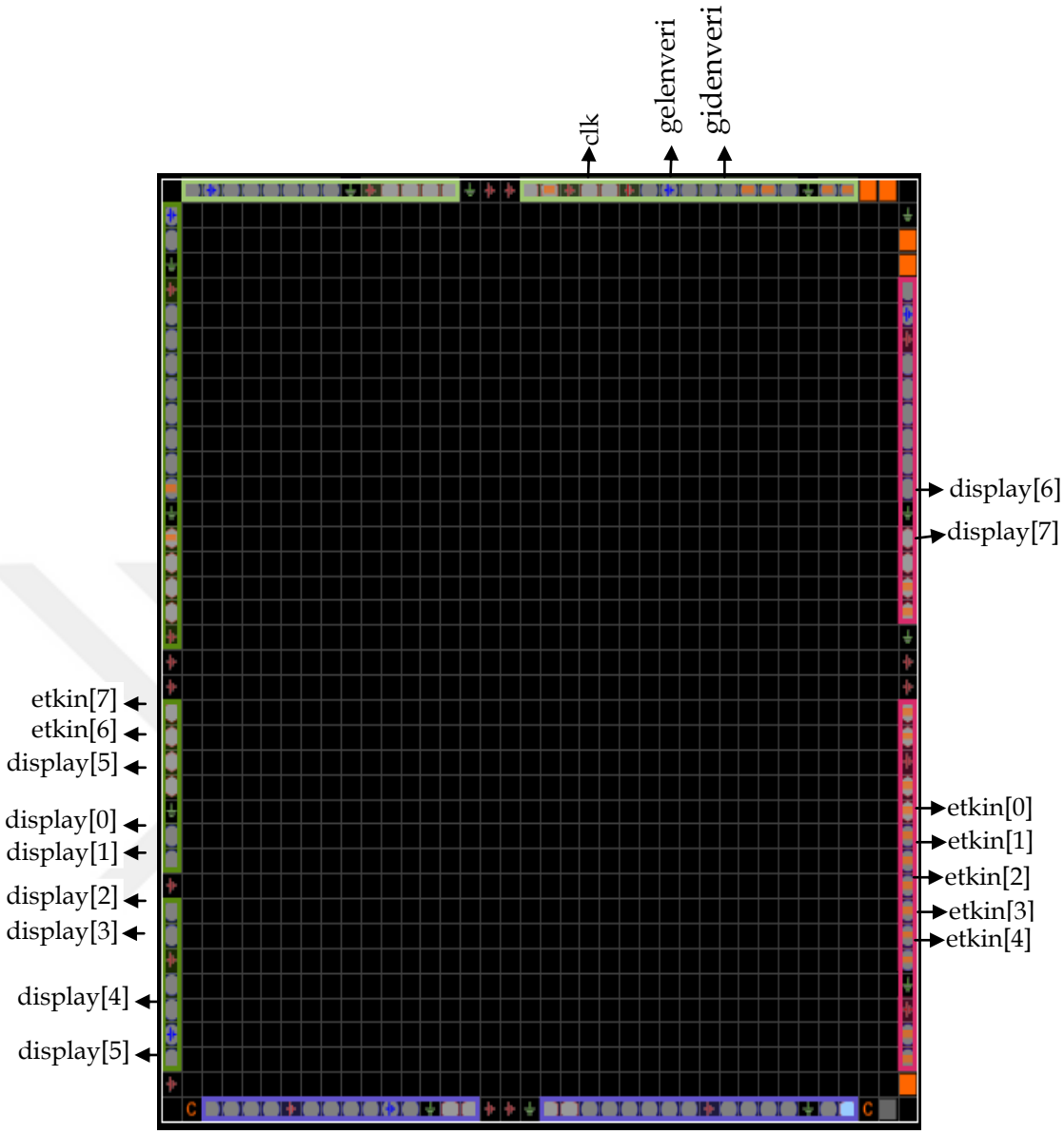
Şekil 4. 8: FPGA güç verileri.

Power Supply

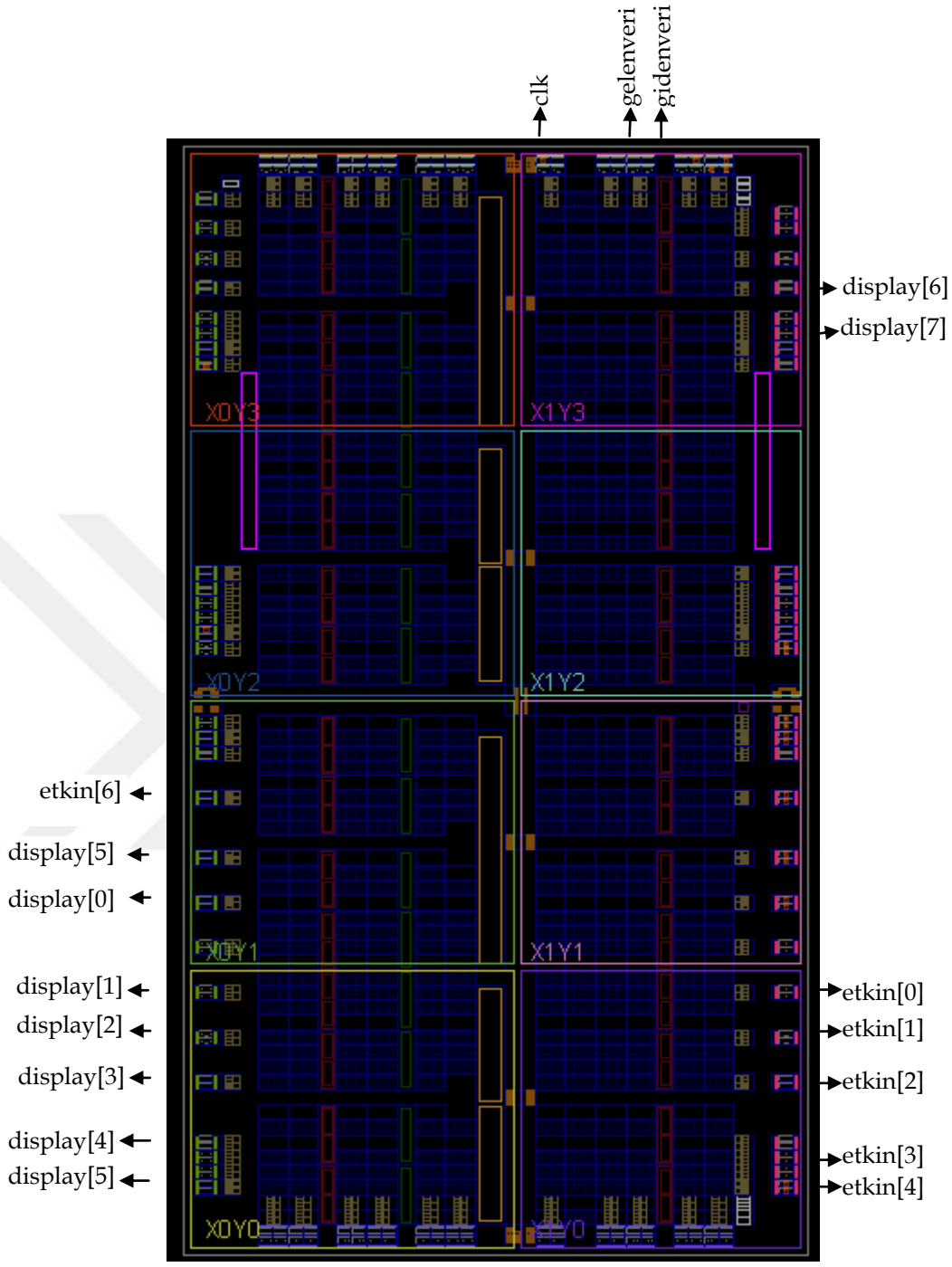
Supply Source	Voltage (V)	Total (mA)	Dynamic (mA)	Static (mA)
Vccint	1.200	8	4	4
Vccaux	2.500	3	0	3
Vcco25	2.500	8	7	1

Şekil 4. 9: FPGA akım gerilim değerleri.

Şekil 4. 10: Gelen verinin osilaskop görüntüsü.

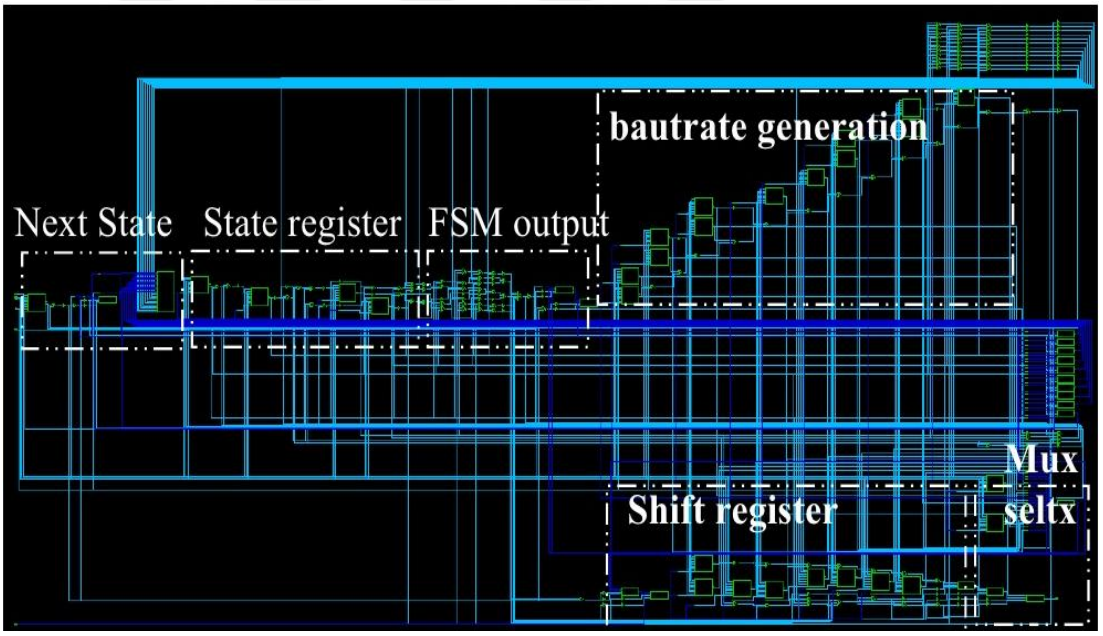


Şekil 4. 11: FPGA pin yerleşim görüntüsü.



Şekil 4. 12: FPGA iç yerleşim görüntüsü.

Şekil 4. 13: Transmitter sentez sonucu RTL.



Şekil 4. 14: Receiver sentez sonucu RTL.

5. SONUÇLAR ve YORUMLAR

Yapılan tez çalışmasında tasarımı yapılan asenkron seri haberleşme devresi ilgili tez çalışmasının ana hedefi olmuştur. Bu tezde öncelikli olarak FPGA üzerinde doğru, kararlı ve esnek çalışabilen bir seri haberleşme protokolü hazırlanmıştır. Bu program ile FPGA'in çalışma frekansına bağlı olarak baudrate oranı istenilen şekilde ayarlanabiliyor. Mikrodenetleyicide ise bu oran donanımın el verdiği şekilde yapılabilmektedir. FPGA'in yapması gereken işler mikrodenetleyici tarafından yapılarak bilgisayar arayüz programı aracılığıyla FPGA'e hazır olarak verilmiştir. Bu şekilde FPGA'in iş yükü ve maaliyeti azaltılmıştır. Seri haberleşme protokolü, mikrodenetleyicilerde internette hazır bir şekilde ücretsiz bulunurken FPGA'in ise kullanıcı tarafından yazılması gerekir. Bu suretle hali hazırda literatüre kazandırılmış olan üzerindeki dış dünya ile veri haberleşmesi için gerekli olan çevresel birimlerin eksiklerinin giderilmesi ilgili tez çalışmasının özgün olmasını sağlamakla birlikte elde edilen bilgi ve tecrübe USB , I2C , Ethernet gibi bilgisayar ve cihaz tabanlı seri haberleşme protokolleri oluşturulması ve gerçeklenmesi açısından önemlidir.

Günümüzde her yönlü kullanım alanı bulunan yeniden programlanabilir bir entegre olan FPGA' in hızlı ve paralel işlem yapma yeteneği her türlü iletişim, kontrol algoritmasının gerçek zamanlı olarak gerçeklenmesinde maliyet ve hız açısından oldukça iyi bir eğim yakaladığı görülmektedir. Bu açıdan asenkron RS232 seri haberleşme devre tasarımının özgün Verilog dili ile yapılmış olması ve gömülü sistemlerin kontrol ve haberleşme yeteneklerinin araştırma amaçlı olarak tasarımı yapılmış sistemin geliştirebilir olmasının da temel bir çalışması olmuştur.

FPGA ve gömülü sistem tasarımlarının akademik ve endüstriyel çalışmalarda hızlı artan bir yer edinmesi yapılan tez çalışmasının önemli ve gelecek vadeden bir çalışma olduğunu göstermekle birlikte gerek lisans gerekse lisans üstü derslerde işlemci, veri haberleşmesi, yazılımsal tabanlı gerçek zamanlı kontrol konularda zevkli ve öğretici bir niteliği olduğu düşünülmektedir.

KAYNAKLAR

[1] Floyd T., (2009), "Digital Fundamentals", 9th Edition, Pearson International Edition.

[2] Yi-yuan F., (2011) "Design and Simulation of UART Serial Communication Module Based on VHDL", 3rd International Workshop on Intelligent Systems and Applications, 978-1, Wuhan, China, 28-29 May.

[3] Web 1, (2017), https://www.allaboutcircuits.com/technical-articles/back-to--the-universal-asynchronous-receiver-transmitter-uart/, (Erişim Tarihi: 24/11/2017).

[4] Jusoh N. F., (2012) "An FPGA Implementation of Shift Converter Block Technique on FIFO for RS232 to Universal Serial Bus Converter", IEEE Control and System Graduate Research Colloquium, 219-224, Shah Alam, Malaysia, 16-17 Jul.

[5] Bhatt D.V., (2012), "Design of A Controller for A Universal Input/Output Port", IEEE International Instrumentation and Measurement Technology Conference, 647 – 652, Graz, Austria, 13-16 May.

[6] Yu S., (2007), "Implementation of a Multi-Channel UART Controller Based on FIFO Technique and FPGA", Second IEEE Conference on Industrial Electronics and Applications, 2633-2638, Harbin, China, 23-25 May.

[7] Paul R., (2012), "Real Time Communication Between Multiple FPGA Systems in Multitasking Environment Using RTOS", Devices, Circuits and Systems International Conference, 1-5, Coimbatore, India, 15-16 March.

[8] Web 2, (2017), http://www.mcu-turkey.com/fpga-ile-seri-haberlesme-rs232-uygulaması, (Erişim Tarihi: 20/01/2017).

[9] Web 3, (2017), http://www.fpganedir.com/FPGA/xilinx.php, (Erişim Tarihi: 20/01/2017).

[10] Web 4, (2017), https://www.fpga4fun.com/SerialInterface4.html, (Erişim Tarihi: 24/11/2017).

[11] Web 5, XILINX Spartan-6 FPGA Board X-SP6-X9 User Manual.Pdf (Erişim Tarihi:24/06/2017)

[12] Abed S., (2012), "Design and Implementation of Interfacing two FPGAs", International Conference on Innovations in Information Technology (IIT), 72-77, Abu Dhabi, United Arab Emirates, 18-20 March.

[13] Görür K., (2014), " Bilgisayar FPGA Tabanli Mikrobilgisayar Mimarisi Kullanılarak Seri Haberleşme Üzerinden Adım Motor Hareket Kontrolü ve Uygulaması", Yüksek Lisans Tezi, Bozok Üniversitesi

[14] Straka M., (2007), "Checker Design for On-line Testing of Xilinx FPGA Communication Protocols," Proceedings 22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems, 152-160, Rome, Italy, 26-28 Septemper.

[15] Erğin S., (2016), "Xilinx Fpga Comport-Usb Üzerinden Senkron Haberleşme Yazılımı Tasarımı", VIII. Bilimsel Kongresi, 1-5, Ankara, Türkiye, 1-3 Eylül.

[16] Barkalov A., (2011), "Optimization Of Circuits Of Compositional Microprogram Control Units İmplemented On Fpga", Cybernetics and Systems Analysis, 47 (1), 166–174.

[17] Awedh M., (2015), "Design and FPGA Implementation of UART Using Microprogrammed Controller", Scholars Journal of Engineering and Technology (SJET), 600-608.

[18] N. R. Laddha, A. P. Thakare, (2015), "Implementation of serial communication using UART with configurable baud rate," International Journal on Recent and Innovation Trends in Computing and Communication, 263-268.

[19] Thakur A., (2013), "Design and Implementation of a BIST Embedded High Speed RS- 422 Utilized UART over FPGA," 4th International Conference on Computing, Communications and Networking Technologies, 1-5, Tiruchengode, India, 10-11 July.

[20] Patel V., (2012), "VHDL Implementation of UART with Status Register", International Conference on Communication System and Network Technologies, 750-754, Gujrat, India, 12-13 May.

[21] Mahat N., (2012), "Design of a 9-bit UART Module Based on Verilog HDL," 10th IEEE International Conference on Semiconductor Electronics, 570-573, Kuala Lumpur, Malaysia, 10 September.

[22] Song K., (2011), "A New Approach to Realize UART," International Conference on Electronic and Mechanical Engineering and Information Technology, 2749-2752, Heilongjiang, China, 12-14 August.

[23] Maimun N., (2005), "The design of high speed UART", Asia Pacific Conference on Applied Electromagnetic, 27-32, Johor, Malaysia, 20-21 May.

[24] Jones S., (1997), "An online testable UART implemented using IFIS", 15th VLSI Test et Astronautic Symposium, 344-349, Monterey, CA, USA, 1 May.

[25] Delvai M., (2002), "Time triggered communication with UARTs", 4th IEEE International Workshop on Factory Communication Systems, 97-104, Vasteras, Sweden, 28-30 August.

ÖZGEÇMİŞ

Yahya TAŞTAN 1986 yılında Erzurum'da doğdu. 2004 yılında başladığı Gazi Üniversitesi Teknik Eğitim Fakültesi Elektronik Öğretmenliği Bölümünü 2008 yılında başarıyla tamamlayarak aynı yıl yüksek lisans eğitimine Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik Mühendisliği Anabilim Dalında başladı. 2009 yılından bu yana meslek liselerinde teknik öğretmen olarak çalışmaktadır.