Gömülü Sistemler İçin Android Tabanlı Bir Mikroişlemci Programlama Yazılımı

Araştırma Makalesi/Research Article

Metin SÜVARİ[1,2], Umut Can ÇABUK[1,3], Yasin YİĞİT[1], Orhan DAĞDEVİREN[1]

[1]Uluslararası Bilgisayar Enstitüsü, Ege Üniversitesi, İzmir, Türkiye
[2]Vestel Elektronik AŞ, Manisa, Türkiye
[3]Elektrik-Elektronik Mühendisliği, Erzincan Üniversitesi, Erzincan, Türkiye
metin_suvari@hotmail.com, ucabuk@erzincan.edu.tr, yasin.yigit@hotmail.com.tr, orhan.dagdeviren@ege.edu.tr
(Geliş/Received:07.08.2018; Kabul/Accepted:26.09.2018)
DOI: 10.17671/gazibtd.451112

***Özet*—** Gömülü sistemler belirli görevleri yerine getirmek amacıyla tasarlanıp beyaz eşya, trafik ışıkları, akıllı fabrikalar, ofis cihazları ve nesnelerin interneti gibi alanlarda kullanılır. Farklı gömülü sistemlerinin programlanması için farklı elektronik kartlar ve bilgisayar desteği gerekebilir. Böyle durumda bazı üreticilerin gömülü sistemlere yazılım yükleme işini dış sağlayıcılardan alınan hazır programlama cihazları ile yaptıkları görülmüştür. Bu cihazlarda paralel programlama imkânı olmadığından yazılım yükleme işi ancak sırayla yapılabilmektedir. Bu cihazlara firma içinde birçok birim aynı anda ihtiyaç duyabilir. Tüm bunlar maliyeti yükseltmekte ve performansı düşürmektedir. Diğer yandan, birçok durumda teknik servis ekipleri müşterilere giderken operasyonel nedenlerle bilgisayar taşımamaktadır. Dolayısıyla, bilgisayar kullanılmadan, tablet veya cep telefonu üzerinden gömülü sistem kartlarına yazılım yükleme işleminin yapılması uygun bir strateji olabilir. Android işletim sistemi günümüzde akıllı telefon ve tabletlerin ötesinde yaygın bir kullanım alanına sahip olmuştur. Öyle ki, gömülü sistemler ve bunları programlamaya yarayan cihazlarda dahi Android kullanılabilmektedir. Tablet ve cep telefonlarının maliyet, kullanım kolaylığı ve taşınabilirlik avantajlarından dolayı kişisel bilgisayarlara göre daha fazla kullanım alanı bulunmaktadır. Bu çalışmada Vestel Beyaz Eşya AŞ için, farklı ürünlere özgü farklı konsept işlemcileri destekleyen, paralel programla yapabilen ve üretim ortamına-koşullarına uygun bir programlama cihazı tasarlanmış ve gerçeklenmiştir. Ayrıca, cihazın kontrol edilmesini sağlayan Windows ve Android uygulamaları da yazılmıştır. Üzerinde çalışılan gömülü sistem beyaz eşya odaklı olmasına karşın, tasarlanan sistem farklı gömülü sistemlerde de etkin olarak kullanılabilir.

***Anahtar Kelimeler*—** android, bellenim, flash programlama, gömülü sistemler, mobil programlama, mikroişlemciler

An Android-based Microprocessor Programmer Software for Embedded Systems

***Abstract*—** Embedded systems are designed to complete certain tasks in the fields of white goods, traffic lights, smart factories, office devices, the Internet of things and many more. Installing programs to different embedded systems may require different integrated circuit cards and/or computer support. Hence, it is observed that some vendors have been outsourcing the devices that are used to program the embedded systems, from sub-contractors. Since most of these devices do not support parallel program installation, the flash programming can only be done sequentially. Yet, many branches in the same company may require these devices contemporaneously. All these increase the costs and repress the performance. Further, in many cases, technical support personnel do not carry computers when visiting customer locations, for various reasons. Hence, it can be a decent strategy that programming cards of embedded systems, without using computers, but only smartphones and tablets. Today, the Android OS has become so widespread, that its usage goes beyond the smartphones and tablets. Yet, the embedded systems and even the devices that are used to program those may use Android. The cost, mobility, ease of use and ease of transport advantages of mobile phones and tablets provide them more areas of use, when compared to personal/office computers. In the case study that constitutes the foundation of our work; a flash programming device that supports different concept processors (specific to different goods), can install (flash) programs in parallel, and complies with the production facility requirements has been designed and implemented for Vestel Beyaz Eşya (White Goods) AŞ, who originally outsources the task. The target field was white goods, but the designed system can easily and efficiently be used in other embedded systems, too.

***Keywords*—** android, firmware, flash programming, embedded systems, mobile programming, microprocessors

1. GİRİŞ (INTRODUCTION)

Mobil teknolojiler geçtiğimiz on yıl içerisinde günlük yaşamımızda vazgeçemediğimiz unsurlar arasına girmiştir. Gelişen teknoloji ve kod yazımının kolaylaşmasının getirdiğini kolaylıkla beraber her türlü ihtiyaca yönelik mobil bir uygulama bulmak mümkündür. Açık kaynak kodlu Android işletim sistemi mobil yazılım sektörünün önde gelen işletim sistemlerinden bir tanesidir. Günlük hayatta birçok çalışma alanında Android cihazlar için geliştirilen çok sayıda uygulama bulunmaktadır. Trafikte telefon kullanımından dolayı yaşanan trafik kazalarının önüne geçmek için geliştirilen bir uygulama olan un-divided kullanıcının izin verdiği kişilerden gelen aramaları kabul ederek ve mesajları okuyarak sürücüye kesintisiz bir sürüş yaşatmayı hedefler [1]. EcoFert isimli Android uygulaması tarım arazilerine atılacak gübre miktarını hesaplayarak maliyetleri düşürmeyi hedeflemektedir [2]. Ekrana bakarken göz hareketlerini algılayan ve bu hareketlerden yola çıkarak sayfa ilerletme, kaydırma işlemlerini yapan bir uygulama olan Pholder Android cihazlar üzerinde çalışmaktadır [3]. Bunların yanı sıra sağlık uygulamaları ve giyilebilir teknolojiler de mobil programlamanın gelişmesiyle birlikte ivme kazanmıştır. Mikroskobik bir cihaz sayesinden telefon kamerasından hücre sayımı gerçekleştirebilen bir uygulama 2017 yılında geliştirilmiştir [4]. Sağlık verilerinin kablosuz ve gerçek zamanlı olarak izleyen bir program Android cihazlar için geliştirilmiştir [5].

Gömülü sistemler genel amaçlı bilgisayarlardan farklı olarak kendileri için önceden tasarlanmış görevleri yerine getirmek için kullanılır. Beyaz eşya, trafik ışıkları, akıllı fabrikalar, ofis cihazları, telsiz duyarga ağları ve nesnelerin interneti gömülü sistemler için temel uygulama alanlarıdır. Gömülü sistemler üzerinde görüntü işleme algoritmaları sayesinde nesne takibi yapılabilmektedir [6], [7]. Yapay sinir ağları ile birlikte kullanılarak hayvan davranışlarını analiz etmek için gömülü sistem programlama kullanılmaktadır [8]. Nesnelerin interneti Endüstri 4.0 ile birlikte ön plana çıkmış olup akıllı fabrikalarda kullanılmaktadır [9]. Telsiz duyarga ağlarında kritik düğümlerin tespiti için geliştirilen algoritma gömülü sistemler üzerinde uygulanmıştır [10].

Günümüzde farklı gömülü sistemlerin programlanması için birçok durumda farklı programlama kartları gerekmektedir. Ayrıca programlama işlemi genelde bir kişisel bilgisayar (PC) yardımıyla yapılmaktadır. Bu yayının derlendiği tez çalışmasına temel teşkil eden projede ise çeşitli mikroişlemcilerin daha hızlı ve pratik bir şekilde programlanabilmesi için bağımsız bir programlama cihazı (Veslink) tasarlanmış ve bu cihazla uyumlu bir Android mobil uygulaması ile bir bilgisayar programı geliştirilmiştir. Mobil uygulama ve bilgisayar programı aracılığıyla, cihaz çalıştırılabilmekte, ayarları yapılabilmekte ve kartlara yazılım yükleme işlemleri gerçekleştirilebilmektedir. Bahsedilen mobil uygulama sayesinde bilgisayar kullanılmadan beyaz eşyalar üzerindeki elektronik kartlara yazılım yüklemek mümkün olmaktadır. Ayrıca, piyasada bulunan diğer (ticari) programlama cihazları sadece belirli marka/model işlemcilere yönelik olarak tasarlanmıştır, bu cihaz ve uygulaması sayesinde önceden tanımı ve ayarları yapılan her türlü işlemci programlanabilmektedir. Ayrıca ileriye dönük olarak yeni işlemci modelleri ihtiyacı oluştuğunda sunucu üzerinden güncelleme de yapılabilmektedir. Lâkin bahsi geçen tez çalışması, ilgili projenin daha ziyade yazılım konularına odaklandığı ve donanım unsurlarını kapsamadığı için, işbu yayında da yazılımsal mimari ve işlevlere odaklanılmıştır.

İlerleyen bölümler şu şekilde organize edilmiştir: Konu hakkındaki bazı yararlı ön bilgiler Bölüm 2'de verilmiştir. Bölüm 3'de ilgili çalışmalar verilmiştir. Bölüm 4'te uygulamayı tasarlamak için kullanılan protokoller tanıtılmıştır. Tasarlanan sistemin mimarisi, programın tanıtılması Bölüm 5'te anlatılmıştır. Tartışma ve sonuçlar ise Bölüm 6'da verilmiştir.

2. ÖN BİLGİLER (PRELIMINARIES)

Mikroişlemciler kendi kalıcı belleği üzerinde yazılı olan program kodlarını çalıştırırlar. Bu programlar bellekte ikilik tabanda veri olarak tutulurlar. Aslında mikroişlemcinin kalıcı belleğine yazılı olan bu kodlar derlenmiş programın işlem kodu (*İng. opcode*) karşılıklarıdır. Bu ikili kodlar işlemci mimarisine bağlı olarak 8, 16, 32 (veya başka) bit genişliğinde kelimelerden (*İng. word*) oluşmaktadır. Her kelime mikroişlemcinin çalışması sırasında komut olarak çalıştırılmaktadır. Onaltılık sayı sistemi kullanılabilirlik açısından çok daha kolay olduğu için, çalıştırılabilir kod, hex olarak bilinen onaltılık sayı sistemindeki sayılardan oluşmaktadır [11]. Derleyiciler tarafından oluşturulan hex kodu, programlayıcı sayesinde işlemcinin kalıcı program veya veri hafızasına yazılmaktadır. Bu işlem, mikroişlemci mimarisine uygun olan haberleşme protokolleri (SWIM, JTAG, SWD, TOOL, SPI vb.) kullanılarak gerçekleştirilmektedir.

Doğrudan işlemcinin çalıştırabileceği hex kodu yazma süreci çok zorlu ve uğraştırıcı olduğu için bir üst seviye programla dili olan çevirici (*İng. assembly*) geliştirilmiştir. Bu sayede kodun derleme işlemi ortaya çıkmış olsa da çok karmaşık olan programlama işi daha basit bir hale gelmiş oldu. Assembly dilinde programlama komutları anlamlı kısaltmalardan oluşmaktadır ve bu komutlar derleyici olarak adlandırılan bilgisayar programı tarafından işlemcinin çalıştırabileceği hex koduna derlenmektedir. Basitlik bu programla dilinin en büyük avantajıdır. Her programlama komutu, mikroişlemci üzerinden bir hafıza alanına denk gelmektedir. Assembly dili işlemci üzerinde gerçekleşen tüm olayların kontrolüne izin verdiğinden günümüzde hâlâ yaygın olarak kullanılmaktadır. Yine de bazı program geliştiricileri günlük yaşamda kullanılan dile benzer programlama dillerini tercih etmektedirler. Bundan dolayı C gibi daha üst seviye programlama dilleri geliştirilmiştir. Bu dillerin en büyük avantajı program geliştirmenin daha kolay ve hızlı bir şekilde geliştirilebilir olmasıdır. Bu dillerde

işlemler derleyiciler tarafından gerekli indirgemeler yapılarak olabilecek en küçük boyuttaki hex koduna dönüştürülmektedir.

```
void McuYazilimGuncelleme() @ ".R_McuYazilimGuncelleme"{
    uint32_t Csm;
    uint8_t * Isrt;

    stMCU_sw_update_typedef MCU_sw_update;

    MSU_Clk_init();

    //Boot Kodu Boyut Kontrol
    if (McuYazilimGuncelleme_Region_end__ < (uint32_t)__McuYazilimGuncelleme__) {
        MSU_IWDG_Reset();
    }

    //Stack Pointer Kontrolu
    Csm = VecTabBas[0]; //Sadece derleyici uyarısını iptal etmek için
    Csm = *(__IO uint32_t *)(__ICFEDIT_intvec_start__);
    if (Csm > __RAM_end__ || Csm < __RAM_start__) {
        __set_MSP(((__RAM_end__ - __RAM_start__) / 2) + __RAM_start__);
    }
    else __set_MSP(Csm);

    MCU_sw_update.HW_ADRES = 1;
    uint32_t err = McuYazilimGuncelleme_KontrolVeGuncelleme(&MCU_sw_update); //HataKodu
    if (err > 0){
        MSU_IWDG_Reset();
    }
```

Şekil 1. C dili ile yazılmış bir mikroişlemci kod bloğu
(A microprocessor code block written with C language)

Şekil 1'de STM32 mikroişlemci için yazılmış örnek bir C kod parçası görülmektedir. Mikroişlemci programlarında kullanılan kütüphane, fonksiyon ve değişkenler bilgisayar programlarından farklı olabilmektedir.

```
    if (GPIO_ReadInputDataBit(HW_ADR1_PORT, HW_ADR1_PIN))  HW_ADRES |= 0x02;
??main_0:
    0x800ec0c: 0xf44f 0x7180  MOV.W     R1, #256                ; 0x100
    0x800ec10: 0xf8df 0x0404  LDR.W     R0, ??DataTable8_7      ; GPIOB_CRL
    0x800ec14: 0xf7f5 0xfe1f  BL        GPIO_ReadInputDataBit   ; 0x8004856
    0x800ec18: 0x2800         CMP       R0, #0
    0x800ec1a: 0xd007         BEQ.N     ??main_1                ; 0x800ec2c
    if (GPIO_ReadInputDataBit(HW_ADR1_PORT, HW_ADR1_PIN))  HW_ADRES |= 0x02;
    0x800ec1c: 0xf8df 0x03f4  LDR.W     R0, ??DataTable8_6      ; 0x20000c47
    0x800ec20: 0x7800         LDRB      R0, [R0]
    0x800ec22: 0xf050 0x0002  ORRS.W    R0, R0, #2
    0x800ec26: 0xf8df 0x13ec  LDR.W     R1, ??DataTable8_6      ; 0x20000c47
    0x800ec2a: 0x7008         STRB      R0, [R1]
    HW_ADRES ++;
??main_1:
    0x800ec2c: 0xf8df 0x03e4  LDR.W     R0, ??DataTable8_6      ; 0x20000c47
    0x800ec30: 0x7800         LDRB      R0, [R0]
```

Şekil 2. Örnek C kodunun Assembly dilinde tercümesi
(Translation of the sample C code to Assembly language)

Şekil 2'de örnek bir C kodu parçasının Assembly dilindeki birebir tercümesi görülmektedir.

```
:020000040800F2
:10000000F813002091EC000879DC00087BDC000884
:1000100089DC000897DC0008A5DC00080000000006F
:10002000000000000000000000000000B3DC000839
:10003000B5DC000800000000B7DC0008B9DC0008EF
:10004000A9EF0008ADEF0008B1EF0008B5EF000818
:10005000B9EF0008BDEF0008C1EF0008C5EF0008C8
```

Şekil 3. Derlenmiş hex kodu
(Compiled hex code)

Şekil 3'te ise mikroişlemciye yüklenecek programın derlenerek işlemcinin kalıcı hafızasına (ROM) yazılacak şekle getirilmiş hali hex kodu formatında gösterilmiştir. Bu hex kodları mikroişlemci tarafından çalıştırılırken ikili (binary) formata dönüştürülürler.

3. İLGİLİ ÇALIŞMALAR (RELATED WORKS)

Mikroişlemci programlamak için kullanılan birçok ticari programlama kartı bulunmaktadır. Renesas marka programlayıcılar ile birlikte gelen Flash Development Toolkit programı ile bilgisayar üzerinden mikroişlemcilere programların yüklenmesi gerçekleştirilebilir. Ayrıca Renesas marka mikrobilgisayarlar için geliştirilen IDE (*İng. Integrated Development Environment*) ile kod yazım ve derlenme işlemleri yapılabilmektedir [12].

Segger firması tarafından geliştirilen J-Link programlama kartları ile birlikte gelen program ile yazılan kodlar komut satırından ilgili mikroişlemciye yüklenebilmektedir. Ayrıca birçok geliştirme ortamına entegre edilebilen bir hata ayıklama aracı da Segger tarafından geliştirilmiştir. J-link programlama kartları birçok ARM ve Renesas mikro işlemcisini desteklemektedir [13].

ARM mikrodenetleyiciler için GP-ARM markalı programlama kartı üretilmiştir. Bu programlayıcı ayırıcı vasıtasıyla 6 adet mikroişlemciyi aynı anda programlayabilmektedir. Firma tarafında Gang-Pro isimli program Windows işletim sistemi için programlama arayüzü olarak geliştirilmiştir [14].

Görsel bir programlama dili olan LabView programlama dilinin gömülü sistemler için üretilen paketleri kullanarak, alt seviye programlar yazılabilmektedir. Donanım üreticileri kendi donanımları için LabView kütüphaneleri oluşturarak kendi donanımları için de görsel programla yapılmasını sağlamaktadır. Yazılan programlar donanım üreticisinin sağlamış olduğu kütüphaneler kullanılarak, mikroişlemciler üzerine yüklenebilmektedir [15].

Bunların yanı sıra ticari olarak geliştirilen Atollic TrueSTUDIO, IAR EWARM, Keil MDK-ARM, ve TASKING VX-toolset gibi araç zincirleri (*İng. toolchain*) de gömülü sistemler için kod geliştirme ortamı, derleyici ve hata ayıklayıcı araçlarını bir paket halinde sunmaktadırlar [16].

Görüldüğü gibi bahsi geçen tüm programlama kartları birlikte gelen programlama yazılımları ile birlikte kullanılabilmektedir ve destekledikleri cihaz türleri ve sayıları çeşitlilik göstermektedir. Ayrıca programlama yapılabilmesi için genelde bir adet bilgisayara ihtiyaç duyulmaktadır.

4. TEKNİK VE TEORİK DAYANAK (TECHNICAL AND THEORETICAL BACKGROUND)

Bu bölümde, çalışmamız kapsamında incelenen seçili mikroişlemcileri programlayabilmek için kullanılması gereken programlama protokollerinden bahsedilecektir.

4.1. SWIM Protokolü (SWIM Protocol)

SWIM (*Single Wire ınterface Module*) adlı haberleşme ve hata ayıklama protokolü, STmicroelectronics firmasının

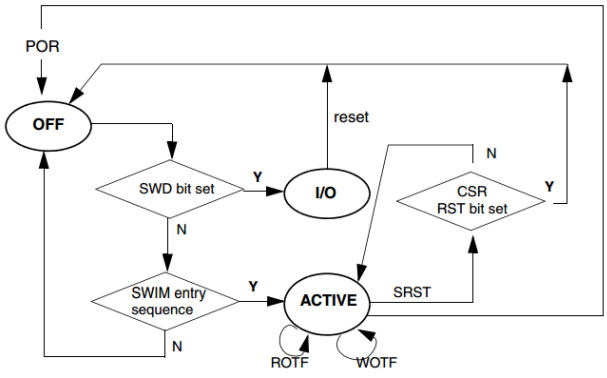
STM8 mikroişlemci ailesi tarafından kullanılmaktadır. STM8 serisi işlemciler için tanımlanan hafıza adres bilgileri ve kaydediciler sayesinde SWIM protokolü ile programlama işlemi gerçekleşmektedir [17].

SWIM haberleşme hattı tek kablo (*İng. single wire*), açık tip (*İng. open-drain*), çift yönlü ve asenkron olarak çalışmaktadır. SWIM, işlemci çalışmasına müdahale etmeden, hata ayıklama amaçlı rastgele erişimli bellek (*RAM*) alanı ve çevresel kaydedicilere (*İng. register*) okuma/yazma olanağı sağlamaktadır [18]. Ayrıca işlemci durdurulduğunda, elektriksel silinebilir programlanabilir salt okunur bellek (*EEPROM*), program hafızası ve işlemcinin hafıza alanına da okuma/yazma imkânı sağlamaktadır. Bunun yanında SWIM protokolü işlemciyi yazılımsal olarak sıfırlama imkânı da sağlamaktadır [19].

4.1.1. SWIM Protokolü Çalışması (Operation of the SWIM Protocol)

SWIM protokolü Şekil 4'te görüldüğü gibi üç farklı durumdan birinde olabilir. İşlemci ilk enerjilendiğinde, SWIM protokolü sıfırlanır ve OFF modu aktif olur. Programlama veya hata ayıklama işlemi gerçekleştirilmek istendiğinde aktif moda getirilmesi gerekmektedir. SWIM'in üç farklı durumu aşağıda belirtildiği gibidir.

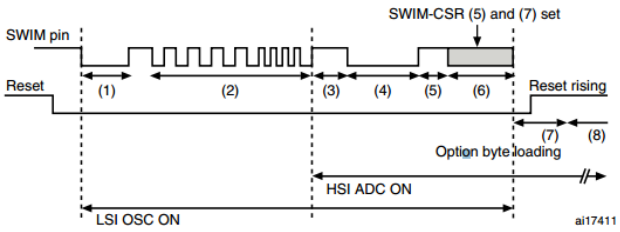
- *OFF*: Bu çalışma modunda SWIM uygulama tarafından giriş/çıkış (I/O) olarak kullanılmaması gerekiyor. İlk olarak SWIM giriş sinyali beklenmektedir. Daha sonra uygulama tarafından I/O çalışma moduna geçebilir.
- *I/O*: Bu moda SWD biti devre dışı bırakılarak uygulama tarafından geçilebilir. Bu modda uygulama bacağı standart I/O olarak kullanabilmektedir. Fakat bu I/O modunda işlemci hata ayıklama imkânı bulunmamaktadır. Ayrıca, işlemci sıfırlandığında SWIM tekrar OFF moda geçecektir.
- *ACTIVE*: Bu moda OFF durumunda belirli sinyal sıralaması geldiğinde geçilmektedir. Bu durumda 3 farklı komut kullanılarak STM8 işlemciler kontrol edilebilmektedir.



Şekil 4. SWIM çalışma durum diyagramı [19].
(SWIM operations state diagram)

4.1.2. Aktifleştirme İşlemleri (Activation Processes)

POR (*Power on Reset*) gerçekleştikten sonra SWIM OFF moduna geçecektir. Bu durumda işlemci içerisindeki LSI osilatör (düşük hızlı saat) otomatik olarak aktifleştirilecektir. Bu sayede ACTIVE moda geçebilmek için gerekli olan SWIM aktifleştirme işlemleri beklenecektir.



Şekil 5. SWIM aktifleştirme işlemleri [19].
(SWIM activation operations)

SWIM'in aktifleştirilmesi için gerekli olan işlemler Şekil 5'te görülmektedir. Adımların detayları aşağıdaki gibidir.

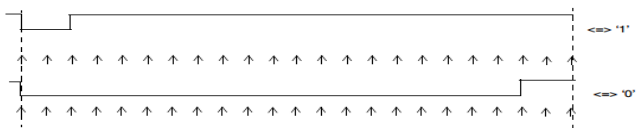
1- SWIM hattı, 16µs (64 pulse yüksek hızlı saat) 0'a çekilmelidir.
2- SWIM hattı 0 seviyesine çekildikten sonra, 4 pulse 1 kHz ve 4 pulse 2 kHz sinyal gönderilmelidir.
3- SWIM giriş sinyalleri düzgün olarak algılandığında, SWIM ACTIVE moda geçer ve HSI (yüksek hızlı saat) osilatörü otomatik olarak aktifleştirilecektir.
4- SWIM giriş işlemi başarılı bir şekilde tamamlandıktan sonra işlemci senkronizasyon sinyali gönderecektir. Senkronizasyon sinyali SWIM hattını 128 x HSI saat sinyali kadar 0 seviyesinde tutacaktır. Bu sayede cihazlar kendi saat sinyallerinin kalibrasyonunu da yapabileceklerdir.
5- SWIM haberleşmesine başlamadan önce en az 300 ns beklenmelidir.
6- SWIM_CSR ye 0xA0h yazılıp, bütün hafıza alanı ve SRST komutu aktifleştirilmelidir.
7- Sıfırlama hattı 1 seviyesine getirilerek, ayar baytı gönderilmelidir. Sonrasında 1 ms stabilizasyon için beklenmelidir.
8- Ayar baytı yükleme ve stabilizasyon tamamlandıktan sonra STM8 HSI frekansı 16 MHz olarak ayarlanır. SWIM saati ise HSI/2 = 8Mhz olur. Bir sonraki bölümde belirtilecek olan bit formatı aktifleştirilir.

4.1.3. Bit Formatı ve Haberleşme Protokolü (Bit Format and Communication Protocol)

Her bit 22 HSI osilatör atışından (pulse) oluşmaktadır. Şekil 6'da gösterildiği üzere;

2 pulse '0' seviyesi ve 20 pulse '1' seviyesi <-> '1'

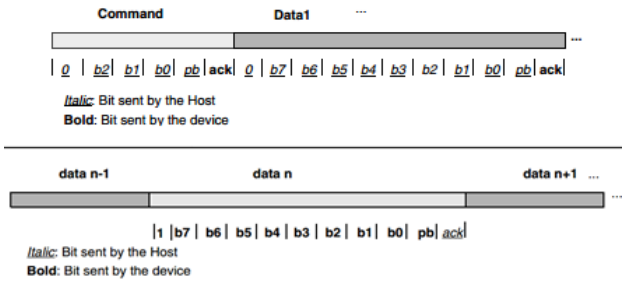
20 pulse '0' seviyesi ve 2 pulse '1' seviyesi <-> '0'



Şekil 6. SWIM bit formatı [19].
(SWIM bit format)

SWIM bit paketi algılanırken hata olmaması için bir seviyesinde en fazla 8 adet sıfır olmalıdır. Sıfır seviyesinde ise en fazla 9 adet sıfır olması gerekmektedir. Eğer bu seviyelerde hata olursa SWIM haberleşmesi sırasında veriler hatalı aktarılacaktır.

ACTIVE modda iletişim ana bilgisayar (*İng. host*) veya programlanacak cihaz tarafından başlatılabilir. Bu yüzden başlık (*İng. header*) içerisindeki bir bit haberleşmeyi kimin başlattığını belirleyecektir. Eğer ana bilgisayar haberleşme başlatacaksa hattın meşgul olmaması gerekmektedir.



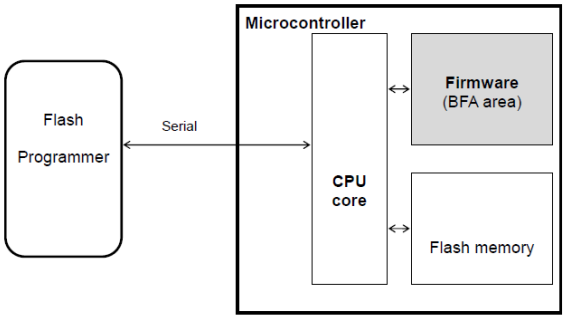
Şekil 7. SWIM komut ve veri formatı [19]
(SWIM command and data format)

SWIM haberleşme protokolünde üç farklı komut kullanılmaktadır. Bu komutların (ROTF, WOTF, SRST) içeriği Şekil 7'de görüldüğü gibi, başlangıç biti veriyi kimin gönderdiğini belirtmektedir. Sonraki üç bit (b2-b0) hangi komut olduğu, sonraki bit komut bitlerinin paritesi ve son olarak da onay biti gönderilmektedir. Toplamda 6 bit komut bilgisi gönderildikten sonra toplamda 11 bit veri paketi gönderilmektedir. Veri paketinde ise, başlangıç biti veriyi kimin gönderdiği, sonraki 8 bit veri ve parite bitleri ve son olarak da onay biti mevcuttur. Komut paketleri ile ilgili ayrıntılı bilgi [19]'da verilmiştir.

STM8 işlemcilerden bazıları uygulama esnasında programlama desteğini de sağlamaktadır [20]. Fakat bütün işlemcilerde bu programlama seçeneği olmaması ve farklı yöntem ve pinler kullanıldığı için bu programlama yöntemi tercih edilmemektedir [21]. STM8 işlemci modellerinde Flash program hafızası ve EEPROM alanı işlemcilerin ailesine ve modellerine göre değişiklik göstermektedir [22]. Bu yüzden yazılım geliştirilirken bütün işlemci modellerini destekleyecek şekilde çalışma yapılmıştır.

4.2. Renesas Flash Programlama (Renesas Flash Programming)

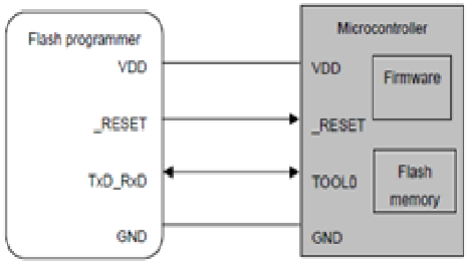
Renesas seri flash programlama, belli komutlar sayesinde mikroişlemci içerisindeki kalıcı program ve veri hafızasına okuma ve yazma imkânı sağlar. Mikroişlemci üretimi anında işlemci üzerine yüklenen açılış kodu (*İng. bootloader*) seri haberleşme olanağı sağlamaktadır. Şekil 8'de mikroişlemcinin genel blok şeması görülmektedir. Merkezi işlem birimi (CPU) çekirdeği boot flash alanındaki yazılımı çalıştırarak, kalıcı hafızaya ulaşabilmeyi sağlar.



Şekil 8. Mikroişlemci mimarisinin blok şeması [12]
(Block diagram of the microprocessor architecture)

4.2.1. İletişim Sistemi ve Fonksiyonel Detaylar (Communication System and Functional Details)

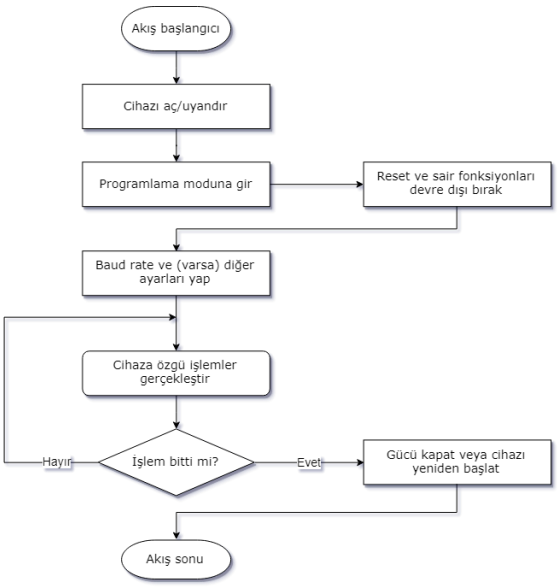
Renesas RL78 serisi işlemcilerde programlama için haberleşme, mikroişlemcinin TOOL0 pini üzerinden tek kablo evrensel asenkron alıcı-verici (UART) iletişimi ile gerçekleşmektedir. Veri haberleşme hızı 115200, 250000, 500000, 1000000 bps olabilmektedir. Veriler 8 bit olarak bir başlangıç biti ve bir stop biti ile birlikte gönderilmektedir. İletişim tek hat üzerinden çift yönlü olduğu için ve iletişim zamanlamasını minimum tutmak için usta-köle (*İng. master-slave*) yapısı kullanılmaktadır. Haberleşme sırasında veri 8 bitlik birimler halinde ve önemsiz bit önce (LSB-first) olacak şekilde gönderilir [23]. Şekil 9'da programlayıcı bağlantı şeması görülmektedir.



Şekil 9. Programlayıcı bağlantı şeması [12]
(Programmer connection diagram)

Flash programlama işleminin düzgün olarak yapılabilmesi için Şekil 10'da gösterilen flash programlama adımları düzgün bir şekilde yapılmalıdır. İlk olarak mikroişlemci

sıfırlanarak programlama moduna girmesi sağlanmalıdır. Bundan sonra haberleşme hızı ayarlanmalıdır. Sonrasında uygun komutlar çalıştırılarak programlama işlemi gerçekleştirilir.



Şekil 10. Flash programlama temel akış diyagramı
(Main flow chart for the flash programming)

4.2.2. Komutlar ve Durum Bilgileri (Commands and State Info)

İşlemci içerisinde yüklü gelen donanım yazılımının (*İng. firmware*) desteklediği komutlar ve açıklamalar Tablo 1'de listelenmiştir. Signature komutuyla mikroişlemcinin program flash ve veri flash alanlarının başlangıç ve bitiş adresleri öğrenilebilmektedir. Security komutlarıyla mikroişlemci içine yazılan koda okuma engellemesi gibi güvenlik önlemleri eklenebilir. Baudrate komutu ile iletişim veri hızı değiştirilebilir. Erase, write ve verify komutları ile de sırasıyla programla işlemi gerçekleştirilmektedir.

Tablo 1. Programlama komutları
(Programming commands)

Kod	Komut	Açıklama
00H	Reset	İletişim senkronizasyon kontrolü için kullanılır.
22H	Block Erase	Tanımlanmış flash alanı içerisindeki verileri siler.
40H	Write	Tanımlanmış flash alanı içerisine alınan verileri yazar.
13H	Verify	Tanımlanmış flash alanı içerisindeki veri ile alınan veriyi karşılaştırır.
32H	Block Blank Check	Flash içerisindeki verinin silinip silinmediğini kontrol eder
9AH	Baud Rate Set	İletişim hızını ve voltaj seviyesini ayarlar.
C0H	Signature	Cihaz bilgilerini okur. (Cihaz ismi, flash adres- alan bilgileri vb.)
A0H	Security Set	Güvenlik bilgilerini değiştirir. (Boot bilgisi, okuma engelleme, vb.)
A1H	Security Get	Güvenlik bilgilerini okur.
A2H	Security Release	Flash opsiyonlarını temizler.
B0H	Checksum	Tanımlanmış flash alanı içerisindeki verinin sağlama bilgisi okunur.

Programlama esnasında oluşabilecek hatalar ve mikroişlemci durum bilgileri Tablo 2'de listelenmiştir. Eğer komut düzgün olarak algılandıysa işlemci komutlara ACK bilgisi dönmektedir. Security komutu ile okunması engellenmiş bir alan okumaya çalışıldığında aşağıdaki tabloda belirtilen koruma hatası alınacaktır. Mikroişlemci tarafından desteklenmeyen bir komut gönderildiğinde ise komut hatası durum bilgisi dönülecektir.

Tablo 2. Durum komutları
(State commands)

Kod	Durum	Açıklama
04H	Komut Hatası	Alınmış desteklenmeyen komut sayısı
05H	Parametre Hatası	Komut içerisindeki parametrelerden birinin uygun olmaması
06H	Normal (ACK)	Normal cevap
07H	Checksum Hatası	Komut içerisindeki verilerin sağlama ile uyumsuz olması durumu
0FH	Doğrulama Hatası	Doğrulama verisinin hatalı olması
10H	Koruma Hatası	Korumalı alan içerisindeki verinin değiştirilmeye çalışılması
15H	Olumsuz ACK	Olumsuz cevap
1AH	Silme Hatası	Silme sırasında hata oluşması
1BH	Silme Kontrol Hatası	Dahili doğrulama ya da silme kontrol hatası
1CH	Yazma Hatası	Yazma sırasında hata oluşması

Komut paketleri gönderilirken Tablo 3'te belirtilen komutlar kullanılmalıdır. Her paket SOH komutu ile başlayıp ETX komutu ile bitmesi gerekmektedir. LEN gönderilecek veri uzunluğu, COM gönderilen komut sayısı, SUM ise gönderilen paketin sağlama (*İng. checksum*) değeri olması gerekmektedir.

Tablo 2. İletişim veri formatları
(Communication data formats)

Kod	Sembol	Açıklama
01H	SOH	Komut paketinin başlangıcı
02H	STX	Veri paketinin başlangıcı
17H	ETB	Blok bilginin bitişi (Son veri paketi değil)
03H	ETX	Bütün verilerin bitişi (Son veri paketi)
-	LEN	Veri uzunluğu (256 için 00H)
-	SUM	Sağlama verisi (1 bayt)
-	COM	Komut sayısı (1 bayt)

Programlayıcı tarafından gönderilen veya alınan komutların formatı Şekil 11'de gösterilmiştir. Programlayıcıdan mikroişlemciye gönderilen komut SOH ile başlar. Sonra gönderilen bayt, veri paketinin uzunluğunu belirtir. COM gönderilecek komut sayısı, SUM ise paketin tamamının sağlama bilgisi olup, paket ETX komutu ile sonlandırılır.

SOH	LEN	COM	Command information (variable length) (Max: 255 bytes)	SUM	ETX
STX	LEN	Data (variable length) (Max: 256 bytes)		SUM	ETB or ETX

Şekil 11. Veri ve komut paketi formatları
(Data and command packets' formats)

4.3. JTAG Programlama (JTAG Programming)

Günümüzde çip tasarımları geliştikçe, geleneksel yollardan elektronik kartların test edilmesi zorlaşmıştır. Özellikle BGA (ball grid array) paket tipindeki mikroişlemcilerde pinler mikroişlemcinin altında kaldığı için yeni metotlar geliştirilmek zorunda kalınmıştır [24]. Sektörün önde gelen firmaları JTAG (Joint Test Action Group) ve sınır tarama (*İng. boundary scan*) IEEE 1149.1 standardını geliştirdiler [25].

JTAG üzerinden yazılım yüklemesi yapılırken JTAG-DP (JTAG Debug Port) kullanılacaktır. Bu kısım IEEE 1149.1 TAP (Test Access Port) ile benzer yapıdadır, fakat sıralı tarama (boundary scan) mimarisi daha çok JTAG ve test amaçlı kullanılmaktadır. JTAG-DP ise hata ayıklama bileşenlerine ulaşmak için kullanılmaktadır. [20]'de JTAG TAP genel mimarisi ve arayüzü verilmiştir. Şekil ile tarif edilen çoklu mimari yapı sayesinde birden fazla JTAG TAP birbirine seri şekilde bağlanabilmektedir. Bu yapı sayesinde hem sıralı tarama yapılabilir hem de IR ve DR register'ları kullanılarak mikroişlemciler için hata ayıklama yapılabilmesine olanak sağlanır [26].

JTAG üzerinden sıralı tarama sadece mikroişlemciler için değil RAM ve flash gibi çok pinli olan entegre devrelerin test edilmesi için de kullanılmaktadır. Elektronik kart üzerindeki bütün JTAG desteği olan entegreler için aynı port üzerinden test yapılabilmektedir [25]. [27]'de ARM hata ayıklama arabiriminin genel blok diyagramı verilmiştir, ilgili tasarım ve projelerde bu yapı dikkate alınmalıdır.

5. SİSTEM MİMARİSİ (SYSTEM ARCHITECTURE)

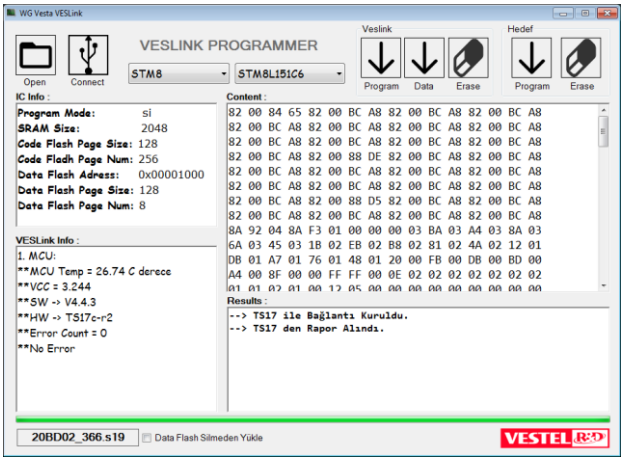
Bu bölümde Veslink programlama cihazının kullanılabilmesi için hazırlanan bilgisayar programı ve Android uygulamasından bahsedilecektir. Şekil 12'de genel görünümü sunulan cihazın donanımsal ayrıntılarına ise bu çalışmada yer verilmemiştir.

Şekil 12. Veslink programlama cihazı
(Veslink programmer device)

Bilgisayar programı Visual Studio ortamı kullanılarak C# dilinde geliştirilmiştir. Bu çalışma kapsamında cihazın kullanımını yaygınlaştırmak ve kolaylaştırmak amacıyla bir Android mobil uygulaması da hazırlanmıştır. Bu Android uygulaması sayesinde ön tanımlı tüm işlemci ve kartlara, bilgisayar gerekmeden, tasarlanan cihaz üzerinden (Veslink) kolayca yazılım yüklemesi yapılabilmektedir. Normalde her üreticinin kendi bilgisayar programı ve kendi programlama cihazının kullanılması gerektiği düşünülürse, önerilen sistem teknik personel için büyük kolaylık getirmektedir.

5.1. Bilgisayar Programı (Computer Program)

Şekil 13'te programlama cihazı Veslink için hazırlanan bilgisayar programının görsel arayüzü verilmiştir. Bilgisayar programı üzerinden yazılım dosyası seçimi, işlemci modeli seçimi ve programlama ile silme işlemleri gerçekleştirilebilmektedir. Arayüz üzerindeki 'Content' bölümü, seçilen yazılım dosyasının içeriğini göstermektedir. 'IC Info' bölümünde ise seçilen işlemcinin program ve veri hafızalarının boyutu ile adres bilgileri görülebilmektedir. 'Veslink Info' bölümünde cihaz donanım yazılımı ve yazılım versiyonları görülmektedir. Son olarak, 'Result' kısmında ise yapılan işlemler ve yükleme sonuçları görülmektedir.



Şekil 13. Bilgisayar programı kullanıcı arayüzü
(Computer program user interface)

4.1.1. Program Dosya Formatları (Program File Formats)

Farklı mikroişlemci derleyicileri (IAR Embedded Workbench, Keil uVision, WICED IDE, ST Visual Develop IDE vb.) farklı formatlarda program dosyaları üretmektedirler [28]. Bu farklılık çoğunlukla üretici tercihlerinden, entelektüel mülkiyet (patent vb.) sınırlamalarından veya hedef mikroişlemcilerde kullanılan teknolojilerin farklılığından kaynaklanmaktadır. Bunlardan biri olan Intel hex formatında her kayıt ASCII olarak ':' (0x3A) ile başlamaktadır. Sonrasında ilgili kaydın ne kadar veri içereceği belirtilmektedir. Veri sayısı bir bayt olduğu için tek satırda en fazla 256 adet veri olabilmektedir. Veri uzunluğu sonrasında iki bayt adres bilgisi mevcuttur. Eğer adres uzunluğu iki bayttan daha

uzun olursa genişletilmiş format tipi kullanılmalıdır. Adres belirtildikten sonra kayıt tipi belirtilmedir. Bunlar '00' veri kayıt, '01' dosya sonu, '02' genişletilmiş adres formatı, '03' bölüm kayıt başlangıcı vb. olabilmektedirler. Son olarak da verinin doğruluğunun kontrolü için sağlama bilgisi yer almaktadır [29].

Tablo 3. Intel hex format [24]

İşaret (':')	Kayıt uzunluğu	Adres	Kayıt tipi	Veri	Sağlama
1 bayt	1 bayt	1 bayt	1bayt	n bayt	1 bayt

Motorola s-record dosya formatı, ikili verinin ASCII hex formatında gösterimi için kullanılmaktadır. Bu dosya farklı uzantılarda (.s19, .mot, .srec vb.) olabilmektedir. Genellikle mikroişlemci ve EPROM'ları programlamak için kullanılmaktadırlar. Srec dosya formatında her kayıt 'S' (0x53) ve sıfırdan dokuza kadar kayıt tipi ile başlamaktadır. Sonrasında adres, veri ve sağlama toplamı kadar bayt sayısı belirtilmektedir. Adres alanı farklı tiplere göre 4, 6, 8 bayt uzunluğunda olabilmektedir. Adres alanından sonra ikinin katları olacak şekilde en fazla 256 adet veri olmaktadır. Son bayt ise sağlama bilgisidir [30]. İkili dosya formatı içerisinde ham veriden başka herhangi bir bilgi (adres vb.) bulunmamaktadır. İkili dosyadan okunacak bilgi direkt olarak mikroişlemcinin program hafızası içine yazılmaktadır.

5.1.2. Mikroişlemci Modelleri (Microprocessor Models)

Programlama cihazı Renesas (RL78, RX63, R78K0, R8C), STMicroelectronics (STM8, STM32), NXP (LPC) firmalarının mikroişlemcileri desteklenecektir. Renesas firması işlemcileri seri programla kullanmaktadır. LPC ve STM32 serisi işlemciler ARM cortex-m3 tabanlı işlemciler oldukları için JTAG programlama ile yazılım yüklenebilmektedir. STM8 serisi işlemcilerde ise SWIM programlama protokolü kullanılmaktadır.

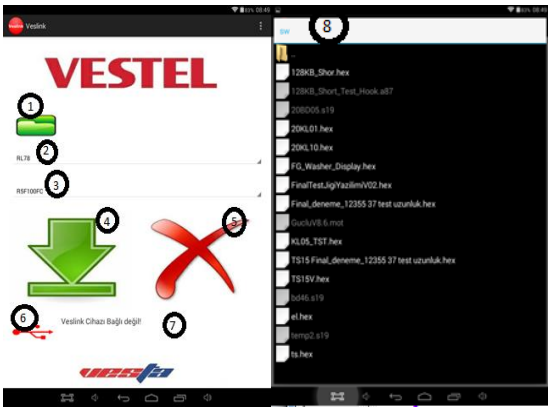
5.2. Android Uygulaması (Android Application)

Android uygulaması, Veslink cihazının kullanıcılar (geliştirici veya servis personeli) tarafından çalıştırılabilmesi, konfigürasyonunun yapılabilmesi ve mikroişlemci programlama işleminin başlatılabilmesini sağlayan sade bir haberleştirici arayüz olarak tasarlanmıştır. Veslink cihazını kullanabilmek için üzerinde bu uygulamayı çalıştıran bir mobil cihaza (ya da bilgisayar programını çalıştıran bir bilgisayara) ihtiyaç vardır. Vestel AŞ personelinin de desteği ile geliştirilen uygulama, Google Play™ yazılım marketi üzerinde genel erişime açık ve ücretsiz olarak, Vestel AŞ tarafından Veslink adıyla yayınlanmıştır.

Android uygulamasının tüm yazım ve geliştirme aşamalarında, Eclipse ADT (Android geliştirici aracı) kullanılmıştır. Java proje yapısı, ana proje ve referans projeyi kapsayan bir tasarım şeklinde oluşturulmuştur. Yazılım tasarımı sürecinde tümleşik geliştirme ortamı olarak kullanılan Eclipse ADT, programcılara verimli bir tasarım oluşturmaları için gerekli araçları sunan bir editör, derleyici, yorumlayıcı ve hata ayıklayıcı olarak nitelendirilebilir.

5.2.1. Uygulama Kullanımı (Use of Application)

Uygulamanın kullanıcı arayüzü Şekil 14'te verilmiş ve kullanımı aşağıda adım adım tarif edilmiştir.



Şekil 14. Android uygulaması ekran görüntüleri
(Android app screenshots)

Dosya aç butonu ile elektronik karta yüklenecek yazılım dosyası seçilmektedir (1 no.lu buton). Yüklenecek dosya "*.hex (Intel formatı) veya *.bin (ikili format)" olabilmektedir (2 no.lu buton). Entegre seçimi kısmında yükleme yapılacak mikroişlemci tipi seçilmektedir (3 no.lu buton). ST, RENESAS mikroişlemci veya SPI seçilebilmektedir. Entegre model seçimi RENESAS işlemciler için R5F100FC ve R4F100JE vb. seçilebilir. SPI için W25Q80BV, W25Q64FV, W25Q128FV gibi modeller bulunmaktadır.

Yükle butonu ile seçilen yazılım programlayıcı içerisine yüklenmektedir (4 no.lu buton). Sil butonu ile programlayıcı içerisindeki yazılımların tamamı silinebilmektedir (5 no.lu buton). Programlayıcıya yazılım yüklerken her seferinde bu butonun kullanılmasına gerek yoktur. Cihaz bağlantı durumu, programlama cihazının bağlı olduğu durumda yeşil yanmaktadır (6 no.lu buton). Bağlı olmadığı durumda ise kırmızı yanmaktadır. Ayrıca bu butona tıklandığında programlama cihazından rapor bilgileri de okunabilmektedir. 7 no.lu kayıt bilgileri hanesinde programlayıcıya bağlı cihaz olup olmadığı ve eğer bağlıysa cihazın güncel durumu hakkında bilgi verilmektedir. Burada programlama sonuçları ve rapor bilgileri de ekrana yazdırılmaktadır. 8 no.lu hanede ise, 1 no.lu butona basınca açılan dosya seçme ekranı ile seçilen klasör ve dosya gösterilmektedir.

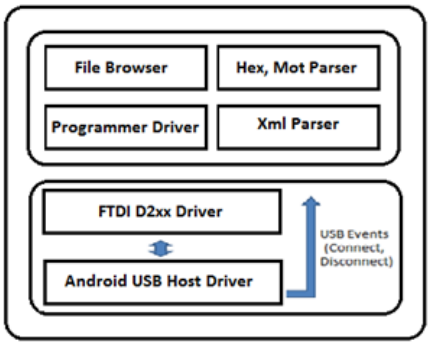
5.2.2. Uygulama Mimarisi (Application Architecture)

Blok şeması Şekil 15'te verilen Android uygulaması iki ana bölümden oluşmaktadır. İlk bölümde Android tarafından desteklenen evrensel seri veri yolu (USB) host sürücüsü bulunmaktadır. Android USB sürücüsünün bir

üst seviyesinde, programlayıcı kartta kullanılan entegre üreticisinin FTDI tarafından hazırlanmış D2xx sürücüsü bulunmaktadır. D2xx sürücüsü ile programlayıcı karta seri olarak veri gönderilip alınabilmektedir. Yine ilk bölümde Android tarafından desteklenen USB bağlantısı ile tetiklenen olaylar sayesinde uygulama otomatik olarak başlatılıp kart bağlantısı yapılabilmektedir.

Android'in kendi dosya seçme seçeneği de kullanılabilmektedir. Bazı telefonlarda dosya açma programı (File Explorer gibi) bulunmadığı için projenin içine dahil edilmesi avantaj sağlayacaktır. Motorola ve hex parser sınıfı sayesinde ise program dosyası programlayıcı karta yüklenecek ham veriye dönüştürülmektedir. XML parser sınıfı, uygulamada seçilen mikroişlemci tipi ve modeline uygun programlama bilgilerini XML dosyaları içinden okumaktadır. XML dosyaları içerisinde, veri hafızası başlangıç ve bitiş adresleri, veri hafızası adresleri, programlama protokolü gibi bilgiler bulunmaktadır. Bu XML dosyaları Android uygulaması ile birlikte oluşturulmaktadır.

Programlayıcı sürücü sınıfı, uygulama ile programlayıcı arasındaki haberleşmeyi sağlamaktadır. Programlayıcı sürücü sınıfı içerisinde okuma, yazma, silme, doğrulama gibi fonksiyonlar bulunmaktadır. Bu fonksiyonlar kullanılarak program dosyasından okunan ham veri, sayfalar (page) halinde programlayıcıya gönderilmektedir. Her sayfa için sırasıyla silme, yazma, doğrulama işlemleri yapılmaktadır. Bu sayede olası hatalı yazılım yükleme işleminin önüne geçilmiştir.



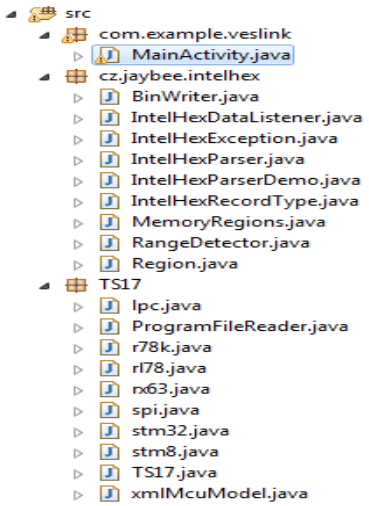
Şekil 15. Android uygulaması blok şeması
(Android app block diagram)

5.2.3. *Programlama Adımları (Steps of Programming)*

Geliştirilen Android uygulaması kullanılırken yapılması gereken işlemler aşağıda sırasıyla belirtilmiştir.

1- Programlayıcı, Android cihaza bağlandığında uygulama otomatik olarak açılır.
2- Dosya seçme butonu kullanılarak program dosyası (.hex, .bin, .s19 …) seçilir.
3- Programlama dosyası seçilmeden önce işlemci tipi ve modelinin uygun olarak seçilmelidir. Bu sayede seçilen program dosyası ve işlemci veri ve veri hafızasının boyutları uyumlu olmadığında uygulama kullanıcıya uyarı verecektir.
4- Programlama başlatıldığında uygulama ilgili program dosyasının içeriğini ve işlemci bilgilerini programlayıcı cihaza gönderir.
5- Daha sonra programlayıcı ve programlanacak kart arasındaki bağlantı yapılır.
6- Son olarak uygulama üzerindeki programlama veya silme butonları kullanılarak yazılım yükleme işlemi gerçekleştirilir.

5.2.4. *Programın Açıklaması (Program Explanation)*



Şekil 16. Android uygulaması sınıflar listesi
(List of classes of the Android app)

Android uygulaması içerisinde bulunan sınıflar Şekil 16'da verilmiştir. TS17 sınıfı FTDI sürücü kütüphanesini kullanmaktadır. FTDI kütüphanesi programlama cihazı ile Android programı arasında USB üzerinden haberleşme sağlamaktadır. Haberleşme FTDI kütüphanesi sayesinde bayt-veri olarak sağlanmaktadır. TS17 sınıfı ise programlama cihazı ile Android arasındaki komut altyapısını sağlamaktadır. Bu komutlar ASCII formatındadır. Bu sayede komutlar izlendiğinde daha anlaşılır olabilmektedir.

IntelHexDataListener sınıfından oluşturulan nesne içerisinde uygun değerlerle listener metodu çağırıldığında dosyadan veri okunur. Tasarım şablonlarında ve modellemede sınıflar tanımlamasını sağlayan soyut sınıf olarak arayüz (interface) kullanılmıştır. Veri ve dosya sonu metotları bu soyut sınıf içerisinde kullanılarak bir şablon çizilmiştir ve binwriter sınıfında bu şablon kullanılmıştır. Birçok sınıf için ortak olan metotları bulunduran bir soyut sınıf olarak IntelHexDataListener'dan bahsetmek mümkündür.

Region sınıfı Compareable arabirimi kullanmıştır. Compareable, kendisini kullanan her sınıfın nesnelerini sıralamaya zorlar. Region sınıfı başlangıç ve bitiş adresleri ve toplam veri uzunluğunu belirleyen veya bilgisine ulaşmayı sağlayan metotları içeren sınıftır. MemoryRegions sınıfı, bütün bölgelerin hafıza adreslerini tutar. Yeni bir bölge ekleme, var olan bölgeyi silme,

toplam bölgelerin başlangıç ve bitiş adreslerini döndüren ve başlangıç adresi ve bitiş adresi eşit olan bölgeleri birleştirme metotlarına sahiptir. Veri okumayı bitirdiğinde bölgeleri sıkıştırır. GetFullRangeRegion metodu bütün bölgeler tek bir bölgeymiş gibi başlangıç ve bitiş adreslerini belirler. IntelHexRecordType enum, Intel hex kayıt tipinin kontrolünü yapar ve değerlerini döner. Eğer kayıt tipi hatalı ise bilinmeyen kayıt tipi olarak 0xFF değerini döndürür. Geri döndürülen veriler, Intel format dokümanı kullanılarak yazılıma eklenmiştir.

IntelHexParser sınıfı, oluşturulduğunda parametre olarak geçilen reader sınıfı oluşturulduysa tekrar oluşturmayı önler. Var olan reader sınıfı kullanılır. Record metodunda hex dosyasından okunan her bir satır tutulmaktadır. Her bir satırdaki kayıt tipi, uzunluğu, adres ve ham veri record metodunu içerir. ParseRecord metodu, ilk olarak satırın iki nokta ile başlamasını kontrol eder. Her satırdaki iki noktadan sonraki uzunluk ikiye bölünerek veri uzunluğu belirlenir ve dizi oluşturulur. Bu dizide ham veri depolanır, sağlama değeri hesaplanır ve dosyadan okunan sağlama değeri kontrol edilir. Ardından toplam veri uzunluğu iki noktadan sonraki ilk değer ile karşılaştırılarak verinin doğruluğu kontrol edilir. Ek olarak büyük ve küçük adresler ve kayıt tipi her bir satır için okunarak record sınıfı oluşturulur. Parse metodu tüm hex dosyasındaki her satır için ParseRecord metodunu çalıştırır. ParseRecord metodu kayıt tipine göre kayıtları işler. Eğer kayıt tipi 0 ise gelen kayıt veri tipindedir. Bu durumda kayıttaki adrese ilgili veriyi yazar. Kayıt tipi 1 ise, bu dosya içerisindeki kayıtların bittiği anlamına gelir. Kayıt tipinin 2 olması genişletilmiş bölüm anlamına gelir ve kayıt içerisindeki veriler genişletilmiş bölümün adresini temsil eder. Bu adres 4 bit sola kaydırılarak sonrasında gelen her adres ile "or" lanır. Böylece adres genişliği bir megabayta kadar arttırılmış olur. Kayıt tipinin 3 olduğu durumda gelen kayıt işlemci hafızasının başlangıç adresidir. Kayıt tipinin 4 olduğu durumda gelen veri genişletilmiş lineer adres tipindedir. Gelen veri 16 bit sola kaydırılarak adres tanım aralığı 32 bite çıkartılmış olur ve adreslenebilir veri büyüklüğü 4 gigabayta kadar arttırılmış olur. Kayıt tipinin 5 olduğu durumda ise gelen veri 32 bit işlemciler için hafıza başlangıç adresidir.

BinWriter sınıfı ise bayt dizisi oluşturarak, her veri eklendiğinde bayt dizisi içerisindeki ilgili adrese veriyi yazar. Tüm bayt dizisi içerisindeki veriler işlemcinin kalıcı hafızasına yazılır.

5.2.5. FTDI Android D2XX Sürücü (FTDI Android D2XX Driver)

FTDI D2xx sürücü farklı işletim sistemleri (Linux, Windows, Android, Mac OS) için FT programlama cihazları ile iletişimde kullanılmak üzere ortak bir uygulama programlama arayüzü sağlamaktadır. Tasarımda, Android uygulamasında FT2232D iletişiminde kullanılmak üzere yazılım paketine eklenmiştir. FTDI sürücüsü için iki farklı uygulama senaryosu çözüm olarak sunulmuştur. İlk çözüm kök yetkisi isteyerek, Android için tüm versiyonları desteklemektedir ve Java yerel arabirimi (JNI) kullanılarak geliştirilmiştir. İkinci çözüm ise USB host desteği için Android Java kütüphaneleri kullanılarak oluşturulmuştur ve Android sürüm 3.2 sonrasındaki versiyonları desteklemektedir. Bu iki çözüm için kullanılan ilk paket kütüphanesi yerel kütüphane (libftd2xx_jni.so) ile derlenmiştir. İkinci paket, Android uygulama arayüzü için d2xx.jar kütüphanesi ile gelmektedir. Bu sayede kolayca Android uygulamasına dahil edilebilmektedir.

FT cihazlarının listelendiği XML listesi ile hedef cihaza (tablet, telefon vs.) bağlanan FT cihazının PID (Product ID), VID (Vendor ID) bilgileri eşleştirilir. Liste içerisinde bulunan cihazlardan biri ile eşleşiyorsa, uygulamayı başlatır. D2xx manager sınıfı FT_Device nesnesinin sağladığı dönüş ile hedef cihazı açmak için API sağlar. FT_Device nesnesi okuma, yazma veya cihaz durumunu kontrol etmek için ilgili bit modu işlemlerini, UART ve EEPROM'u işlemektedir. Cihaza artık ihtiyaç kalmadığında, FT_Device objesi close() metodu ile kapatılabilir. D2xx arayüzü standart işletim sisteminde var olmayan haberleşme işlemlerinde farklı modlarda özel fonksiyonları da desteklemektedir.

D2xx manager sınıfında desteklenen fonksiyonlardan çalışmada kullanılan bazı fonksiyonlar Tablo 5'teki gibidir. Ayrıca aşağıdakilere ek olarak, portu sıfırlamak, yeniden yüklemek ve taramak, EEPROM'a yazmayı, okumayı, silmeyi, programlamayı, kullanıcı alan boyutunu okumayı ve buraya yazmayı sağlayan fonksiyonları da içermektedir.

Tablo 4. FTDI sürücü fonksiyonları
(FDDI driver functions)

setVIDPID	Cihaz için ID ataması yapılmasını sağlar.
getVIDPID	Cihaz için ID sorgulamasını sağlar.
getDeviceInfoList	Bağlı cihazları listeler.
Open	ID'ye göre cihaz iletişimini açar. Varsayılan 0'dır.
Close	Açık cihazın iletişimini kapamak için kullanılır.
Read	Verileri cihazdan okumak içim kullanılır.
Write	Verileri cihaza yazar.
setBaudRate	Veri hızını ayarlamak amacıyla kullanılır.
setDivisor	Veri hızını standart olmayan veri hızında bir değere ayarlamak için kullanılır.
setDataCharachteristic	Bit sayısı, durdurma bitleri, parite gibi ayarları yapmak için kullanılır.
setTimeouts	Okuma ve yazma için geçerli zaman aşımı sürelerini ayarlamak amacıyla kullanılır.
setDtr	Veri hazır sinyalini kontrol etmek amacıyla kullanılır.
clrDtr	Veri hazır sinyalini temizlemek amacıyla kullanılır.
setRts	Veri göndermeden önce yapılan sorgudur.

6. TARTIŞMA VE SONUÇ (DISCUSSION AND RESULTS)

Günümüzde farklı üreticilere ait (veya aynı üreticiye ait fakat farklı model) mikroişlemcileri programlamak için farklı elektronik cihazlar gerekmektedir. Üstelik bu cihazları kullanabilmek için de farklı bilgisayar programları gerekmektedir. Çeşitli form ve modellerde temin edilen gömülü işlemci programlama cihazları; tüketici elektroniği ve beyaz eşya sektöründe faaliyet gösteren kuruluşlarda, üretim, ar-ge, test grupları ve de teknik servisler tarafından yaygın olarak kullanıldığı için önemli bir maliyet kalemi oluşturmaktadır. Örneğin, sadece yayında bahsi geçen projenin gerçekleştirildiği Vestel AŞ Beyaz Eşya tesislerinde üretilen veya bakımı yapılan farklı ürünlerde 10 farklı konsept işlemci kullanılmaktadır. Dolayısıyla tesislerde görev alan sorumlu personelin gerekli hallerde her bir işlemci için ayrı bir programlama cihazı ve onunla uyumlu bir bilgisayar programı kullanması gerekmektedir. Bu durum, fazla malzeme gereksinimi ve yüksek öğrenme eğrisi nedeniyle özellikle sahada ve müşteri lokasyonunda destek veren teknik servis personeli için büyük zorluk oluşturmaktadır. Çalışmamız kapsamında üretilen çok fonksiyonlu Android tabanlı programlama ekipmanı (uygulamayı çalıştıran mobil cihaz ve Veslink cihazı), tek başına tüm mikroişlemciler için ve bilgisayara da ihtiyaç duymadan programlama imkânı sağlamaktadır. Böylece hem bu maliyetlerin azaltılması sağlanacak hem servislerin etkinliğini arttıracak hem de firmalara özgü basit kullanıcı ara yüzü ve farklı işlemci desteği sayesinde firmaların rekabet gücünü arttıracaktır. Ekipman, bu iş için özel olarak geliştirilen yazılımı ihtiva eden herhangi bir uyumlu mobil cihaz (telefon, tablet vs.) ile çalıştırılabilir. Bu çalışmada tasarlanan Android yazılımının ve kullanılan tekniklerin avantajları aşağıda listelenmiştir.

- Uygulama bir Android uygulaması olduğu için, pek çok marka/model tablet ve telefonla birlikte kullanılabilecektir.
- Piyasada kullanılan mevcut ürünler sadece belirli işlemcileri desteklemektedir. Bu cihaz sayesinde farklı konsept işlemciler tek bir cihaz ile programlanabilecektir.
- Uygulama en son kullanılan program dosyasını kendi hafızasına da yazarak bilgisayardan bağımsız yükleme işlemi gerçekleştirilmesini sağlayacaktır.
- İleriye yönelik yeni işlemci desteği istendiğinde, yerel sunucu üzerinden otomatik donanım yazılımı ve bilgisayar yazılımı güncellemesi yapılabilecektir.
- Donanımsal ve yazılımsal bütün ar-ge çalışması sırasında yurtiçi ve yurtdışı başka firmalardan hazır çözüm alınmadığı için geliştirme ve eklentiye açıktır.

Son olarak, bu çalışmanın devamı niteliğinde geleceğe yönelik planladığımız çalışmaları aşağıda listeliyoruz.

- Derleyici ve hata ayıklama desteği (henüz) bulunmamaktadır. Bu önemli özellik eklenecektir.
- Ağ üzerinden otomatik cihaz ve bilgisayar yazılımı güncelleme desteği olduğu için ayrıca güvenlik önlemleri alınması da gerekmektedir.
- Sürekli yeni işlemciler eklenmek istenirse, bunları destekleme ihtiyacı nedeniyle geliştiriciler için sürekli bir iş yükü oluşturacaktır. İş yükünü hafifletmek için açık bir veritabanı düşünülmektedir.
- Nesnelerin interneti kapsamındaki [31-34] teknolojik ilerlemelere uygun geliştirmeler yapılması ön görülmektedir.

BİLGİLENDİRME (ACKNOWLEDGMENTS)

Bu çalışma, TÜBİTAK - Teknoloji ve Yenilik Destek Programları Başkanlığı (TEYDEB) tarafından, 3150940 no.lu proje kapsamında desteklenmiştir. Ayrıca, yazarlar Vestel Elektronik San. ve Tic. A.Ş. Test Geliştirme Mühendisliği ekibine katkıları için teşekkürlerini sunar.

KAYNAKLAR (REFERENCES)

[1] V. E. C. Punay, H. S. Briones, J. C. L. De Leon, J. E. Petralba, "unDivided: An android application for anti-distracted driving," **in 2017 International Conference on Orange Technologies (ICOT)**, 111–114, 2017.

[2] M. V Bueno-Delgado, J. M. Molina-Martínez, R. Correoso-Campillo, P. Pavón-Mariño, "Ecofert: An Android application for the optimization of fertilizer cost in fertigation," *Comput. Electron. Agric.*, 121, 32–42, 2016.

[3] C. Wirawan, H. Qingyao, L. Yi, S. Yean, B.-S. Lee, F. Ran, "Pholder: An Eye-Gaze Assisted Reading Application on Android," **in 2017 13th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)**, 350–353, 2017.

[4] Y. Zeng et al., "A low cost and portable smartphone microscopic device for cell counting," *Sensors Actuators A Phys.*, 274, 57–63, 2018.

[5] D. Lou et al., "A Wireless Health Monitoring System based on Android Operating System," *IERI Procedia*, 4, 208–215, 2013.

[6] T. Öztürk, Y. Albayrak, Ö. Polat, "Object tracking by PI control and image processing on embedded systems," **in 2015 23nd Signal Processing and Communications Applications Conference (SIU)**, 2178–2181, 2015.

[7] O. Elkeelany, S. Moosa, "Towards real-time embeded system design for video capturing and privacy protection," **in 2016 Future Technologies Conference (FTC)**, 593–599, 2016.

[8] D. Gutierrez-Galan et al., "Embedded neural network for real-time animal behavior classification," *Neurocomputing*, 272, 17–26, 2018.

[9] R. Y. Zhong, X. Xu, L. Wang, "IoT-enabled Smart Factory Visibility and Traceability Using Laser-scanners," *Procedia Manuf.*, 10, 1–14, 2017.

[10] O. Dagdeviren, V. K. Akram, "An energy-efficient distributed cut vertex detection algorithm for wireless sensor networks," *Comput. J.*, 57(12), 1852–1869, 2013.

[11] T. Zhe, Development and Application Tutorial of Embedded Systems[M], 2005.

[12] R. E. Corporation, R. E. Corporation, R. E. Corporation, "E1 Emulator R0E000010KCE00 E20 Emulator R0E000200KCT00 User' s Manual," no. May, 2014.

[13] Segger, "J-Link / J-Trace User Guide," 2015.

[14] Gang-Pro "GP-ARM Product Details," 2000.

[15] C. Elliott, V. Vijayakumar, W. Zink, R. Hansen, "National Instruments LabVIEW: A Programming Environment for Laboratory Automation and Measurement," *J. Lab. Autom.*, 12(1), 17–24, 2007.

[16] UM1451 User Manual, "User manual Getting started with software development toolchains for the STM32L-DISCOVERY board," September 2011, pp. 1–30.

[17] ST MicroElectronics, STM8S Series and STM8AF Series 8-bit microcontrollers RM0016 Reference manual [Z], Doc ID 14587 Rev 14, Geneva, Switzerland, 2017.

[18] STM8 in-application programming (IAP) using a customized user-bootloader AN2659 Application note [Z], Doc ID 14153 Rev 4, Geneva, Switzerland, 2012.

[19] STM8 SWIM Communication Protocol and Debug Module UM0470 User manual, DocID14024 Rev 4, Geneva, Switzerland, 2016.

[20] ST MicroElectronics, STM8 CPU Programming Manual Technical Data[Z], Doc ID 13590, Rev 3, Geneva, Switzerland, 2011.

[21] How to program STM8S and STM8A Flash program memory and data EEPROM [Z], Doc ID 14614 Rev 3, Geneva, Switzerland, 2011.

[22] L. Zhao, W. Xie, G. Wu, X. Zhang, In-circuit programmer of STM8 based on MCU, **In Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on**, vol. 8, 4273-4276, IEEE, 2011.

[23] W. Liu, W. G. Zhang, Design and Implementation for an ISP Downloading Unit [J]. Microprocessors, 6, 033, 2010.

[24] T. Martin, **The Designer's Guide to the Cortex-M Processor Family: A Tutorial Approach**, First Edition, Newnes, 2013.

[25] C. M. Maunder, R. E. Tulloss, **The test access port and boundary scan architecture**, IEEE Computer Society Press, Los Alamltos/Washington DC, USA, 242-247, 1990.

[26] Internet: IEEE Std 1149.1-2001- IEEE Standard Test Access Port and Boundary Scan Architecture, https://standards.ieee.org/findstds/standard/1149.1-2001.html, 27.11.2016, 21:53

[27] J. Yiu, **The Definitive Guide to ARM® Cortex®-M3 and Cortex®-M4 Processors**, Third Edition, Newnes, 2013.

[28] M. Verle, **PIC Microcontrollers**, First Edition, Mikroelektronika, 2008.

[29] Internet: Intel Hexadecimal Object File Format Specification Rev. A, 1/6/88. [Online], http://www.interlog.com/~speff/usefulinfo/Hexfrmt.pdf, 23.07.2018.

[30] Internet: MOTOROLA M68000 FAMILY Programmer's Reference Manual (Includes CPU32 Instructions). [Online] https://www.nxp.com/docs/en/reference-manual/M68000PRM.pdf, 14.03.2018.

[31] L. Atzori, A. Iera, G. Morabito, "The Internet of Things: A survey," *Comput. Networks*, 54(15), 2787–2805, 2010.

[32] O. Dağdeviren, V. K. Akram, "TinyOS Tabanlı Telsiz Duyarga Ağları için Bir Konumlandırma ve k-Bağlılık Denetleme Sistemi", *Bilişim Teknol. Derg.*, 10(2), 139–152, 2017.

[33] M. Özarslan Yatak, B. Göktaş, F. Duran, "Design and Implementation of Android-Based Autonomous Human Tracking Vehicle", *Bilişim Teknol. Derg.*, 11(2), 157–162, 2018.

[34] M. Dener, "Kablosuz Sensör Ağlar ile Yeri Tespit Edilen Doktorların Konum Bilgilerinin Android ve Web Tabanlı Platformlar Üzerinden Görüntülenmesi", *Bilişim Teknol. Derg.*, 11(2), 203–210, 2018.