4 ANALİZ VE MODELLEME

Bu projede tek kullanımlık parolalar üreten bir sistem tasarlanmıştır. Tek kullanımlık parolalar üretmek için 3-DES algoritması kullanılmıştır. Proje şifreleme algoritması ve oluşturulan parolaları gösterecek gösterge donanımı bileşenlerinden oluşur.

Projenin en önemli parçası şifreleme algoritmasıdır. Şifreleme algoritması mikroişlemci içine yerleştirilecek ve tek kullanımlık parolalar üretmek için kullanılacak kodları içerir. Bu kodlar bir ana program, bir şifreleme altprogramı ve bir de parola çözme altprogramından oluşur. Ana program içinde gerek duyulduğunda altprogramlar çağrılır ve böylece şifreleme veya parola çözme işlemleri gerçekleştirilir. 3-DES algoritması DES algoritmasının özel bir halidir. 3-DES algoritmasında iki adet 56 bitlik anahtar kullanılır. İlk anahtar verilen bir mesajı şifrelemek için kullanılır. İkinci anahtar ise şifrelenen mesajı çözmek için kullanılır. İkinci anahtar birinci anahtardan farklı olduğu için ikinci işlemde parola çözme işlemi yerine tekrar şifreleme işlemi gerçekleştirilmiş olur. Böylece verilen mesaj art arda iki kez şifrelenmiş olur. Son olarak mesaj birinci anahtar kullanılarak tekrar şifrelenir ve böylece başlangıçta elimizde bulunan mesaj art arda üç kez şifrelenmiş olur. 3-DES algoritması bu şekilde çalışır. Bu nedenle algoritma ana program, şifreleme altprogramı ve parola çözme altprogramından oluşacak şekilde tasarlanmıştır.

Gösterge donanımı olarak HY-1602 LCD gösterge seçilmiştir. Bu göstergede içine veri yazılabilen bellekler vardır. Bu LCD göstergenin çalışması için gerekli ayarlamalar öncelikle yazılım ile yapılmalıdır, daha sonra LCD göstergeye gönderilen veriler ekranda gözükürler. LCD göstergeye tekrar veri gönderene kadar göstergenin belleğine bir önceki yazılmış veriler ekranda kalmaya devam ederler. Tek kullanımlık parola üreten ana program buna uygun bir şekilde tasarlanmıştır.

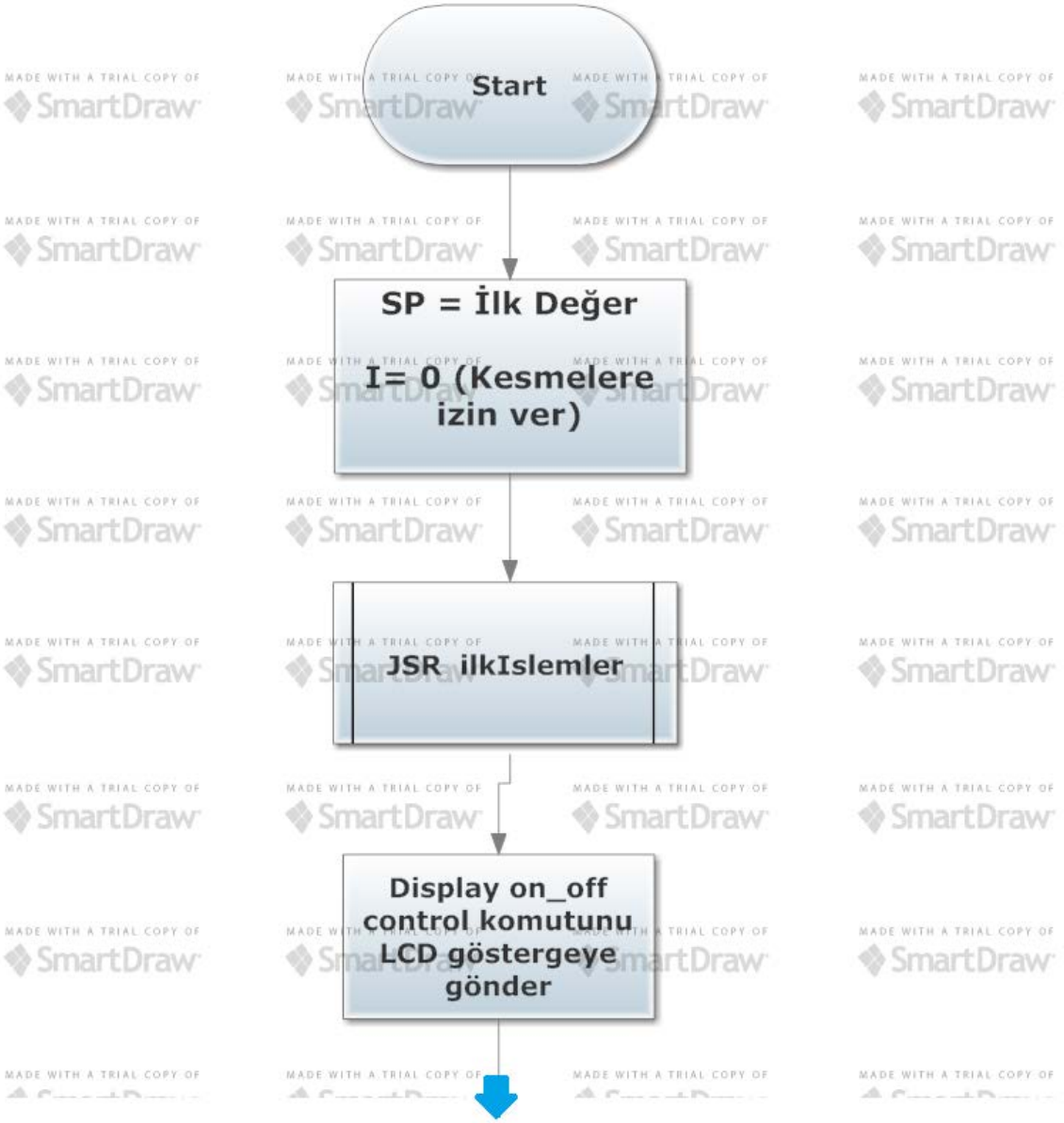
Yazılım tasarımında öncelikle DES algoritmasının kullanacağı sabit tablolar ROM bellek adreslerinde tanımlanmıştır. Daha sonra algoritmanın kullanacağı geçici veriler RAM bellek adreslerinde tanımlanır. Burada en önemli RAM bellek değişkeni şifrelenmiş mesaj değişkenidir. Bu değişken başlangıçta şifrelenecek ilk veriyi tutar. Daha sonra 3-DES algoritması her çalıştırıldığında bu değişken üzerine yeni bir parola yazılır.

Aşağıda Şekil 4.1'de şifrelenmiş mesaj değişkeni görülmektedir. Bu değişken DES şifreleme ve DES şifre çözme altprogramları tarafından kullanılır ve bu değişkene üretilen parolalar yazılır.

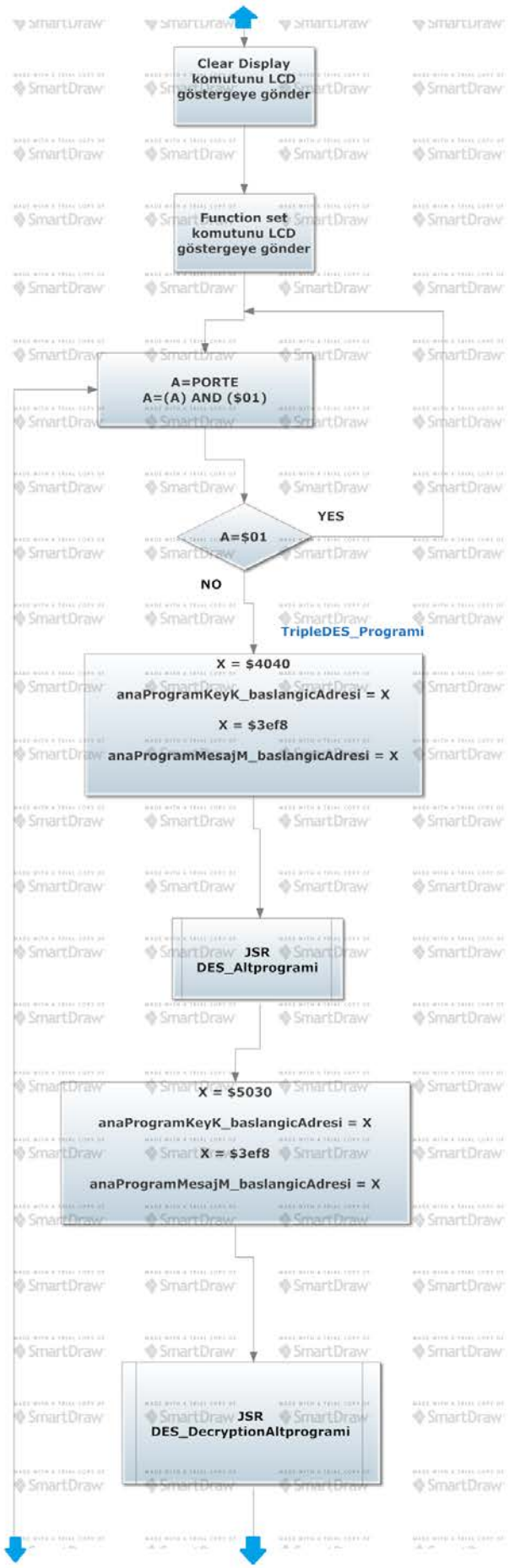
```
;*************************************************************************************
;* Sifrelenmis mesaj SifrelenmisC olarak gosterilir ve bu veri
;* bellekte $0ef8 ve $0f37 adresleri arasinda bulunur($0f37 adresinde son eleman vardir)
;*************************************************************************************

              ORG $3EF8
SifrelenmisC_veriAdresi    DC.B $00,$00,$00,$00,$00,$00,$00,$01
                           DC.B $00,$00,$01,$00,$00,$00,$01,$01
                           DC.B $00,$01,$00,$00,$00,$01,$00,$01
                           DC.B $00,$01,$01,$00,$00,$01,$01,$01
                           DC.B $01,$00,$00,$00,$01,$00,$00,$01
                           DC.B $01,$00,$01,$00,$01,$00,$01,$01
                           DC.B $01,$01,$00,$00,$01,$01,$00,$01
                           DC.B $01,$01,$01,$00,$01,$01,$01,$01
```

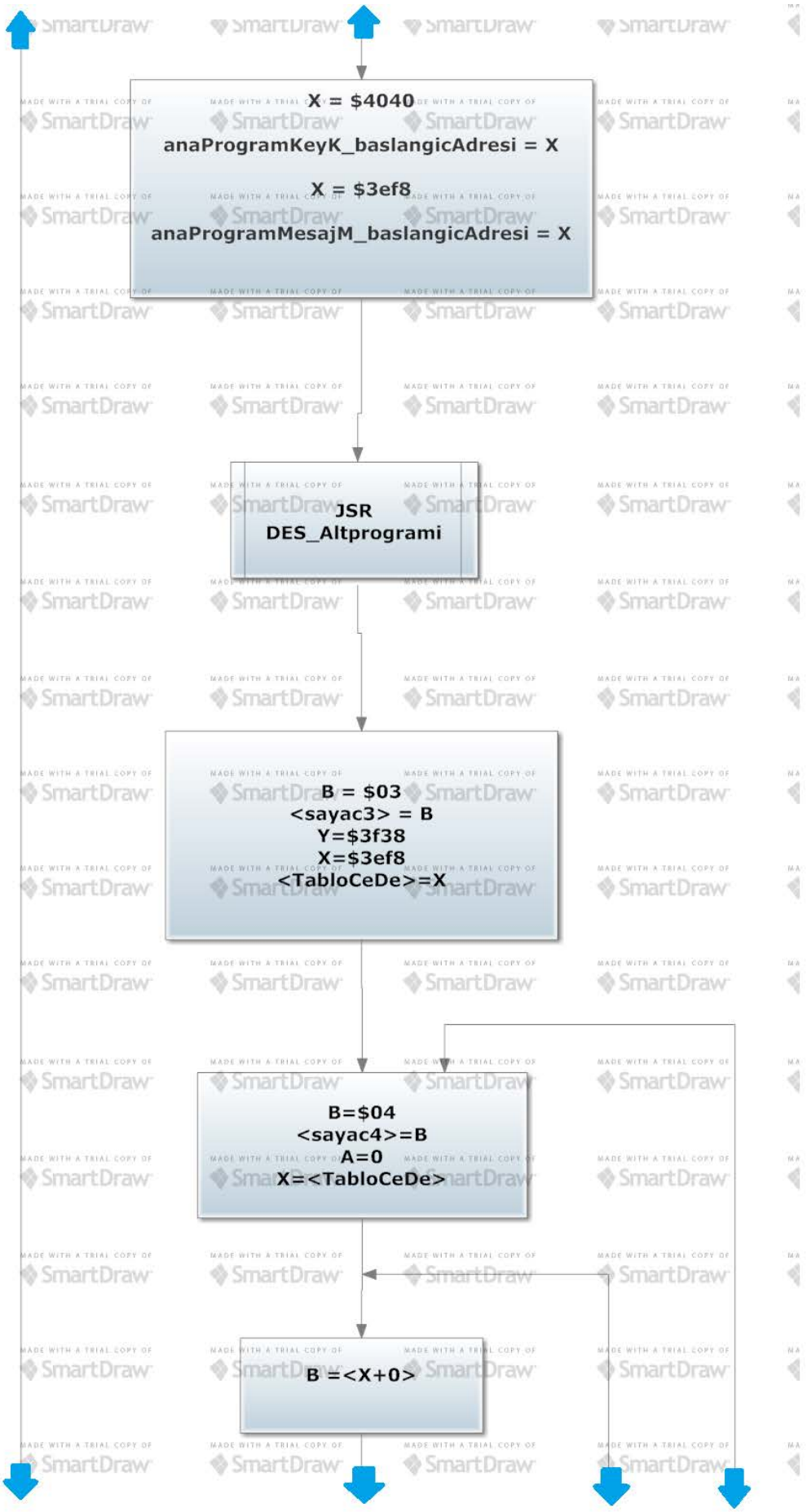
Şekil 4.1: Şifrelenmiş Mesaj Değişkeni

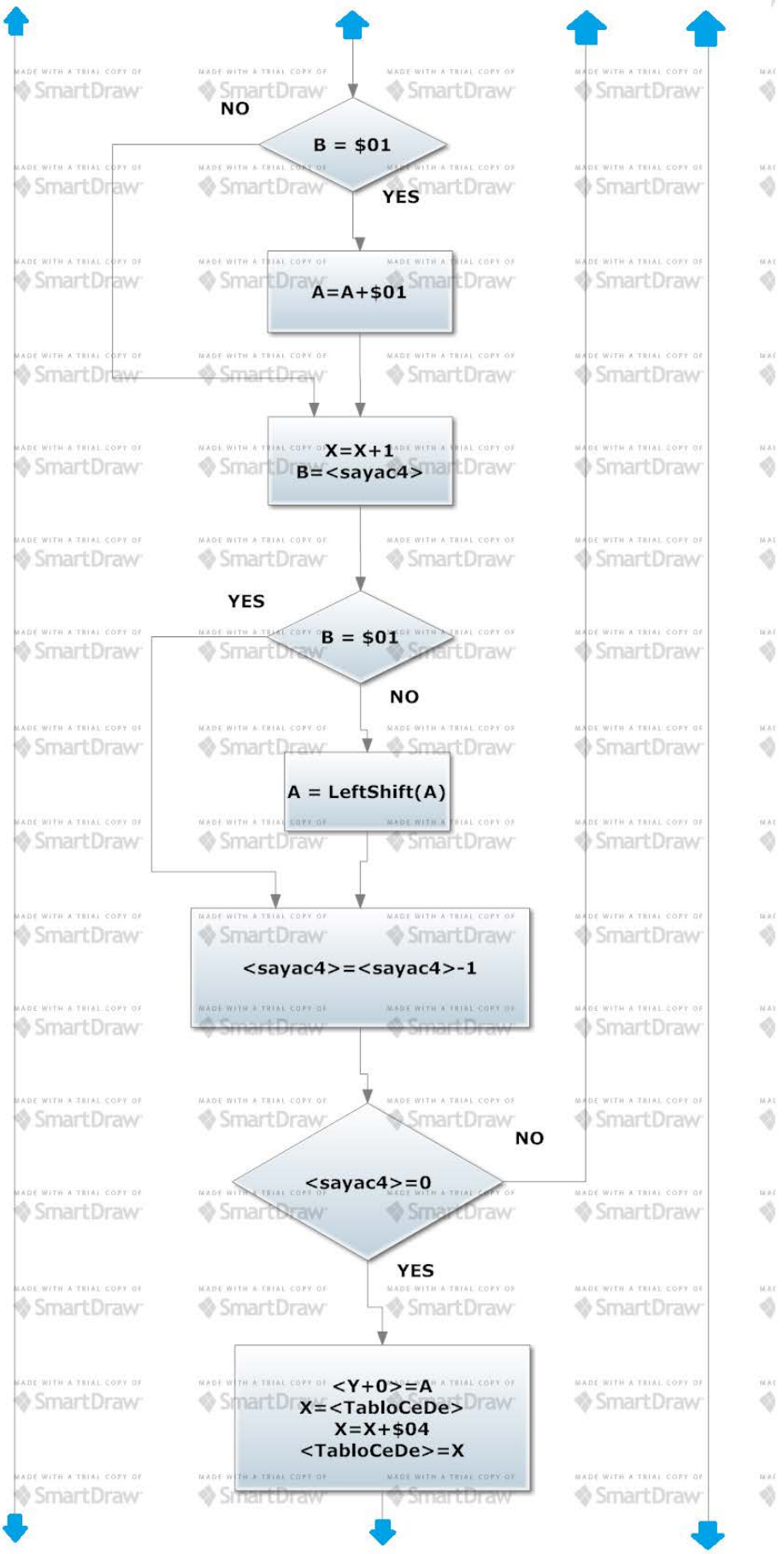
Aşağıda tek kullanımlık parola üreten sistemin yazılımı ile ilgili olan akış şemaları verilmiştir. Bu projede yazılım bir ana program, bir şifreleme altprogramı ve bir de parola çözme altprogramından oluşur. Ana program içinde gerek duyulduğunda altprogramlar çağrılır ve böylece şifreleme veya parola çözme işlemleri gerçekleştirilir. 3-DES algoritması DES algoritmasının özel bir halidir. 3-DES algoritmasında iki adet 56 bitlik anahtar kullanılır. İlk anahtar verilen bir mesajı şifrelemek için kullanılır. İkinci anahtar ise şifrelenen mesajı çözmek için kullanılır. İkinci anahtar birinci anahtardan farklı olduğu için ikinci işlemde parola çözme işlemi yerine tekrar şifreleme işlemi gerçekleştirilmiş olur. Böylece verilen mesaj art arda iki kez şifrelenmiş olur. Son olarak mesaj birinci anahtar kullanılarak tekrar şifrelenir ve böylece başlangıçta elimizde bulunan mesaj art arda üç kez şifrelenmiş olur. 3-DES algoritması bu şekilde çalışır. Bu nedenle algoritma ana program, şifreleme altprogramı ve parola çözme altprogramından oluşacak şekilde tasarlanmıştır.

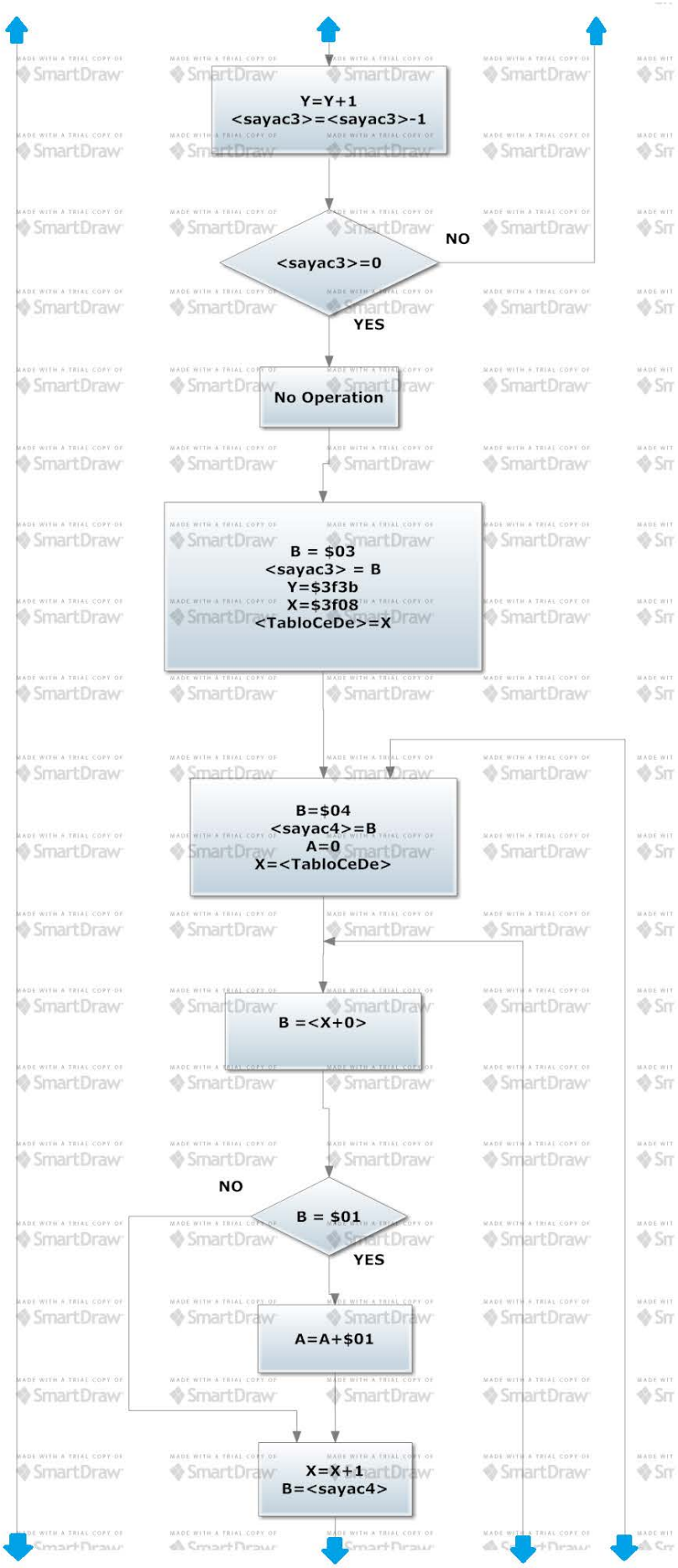


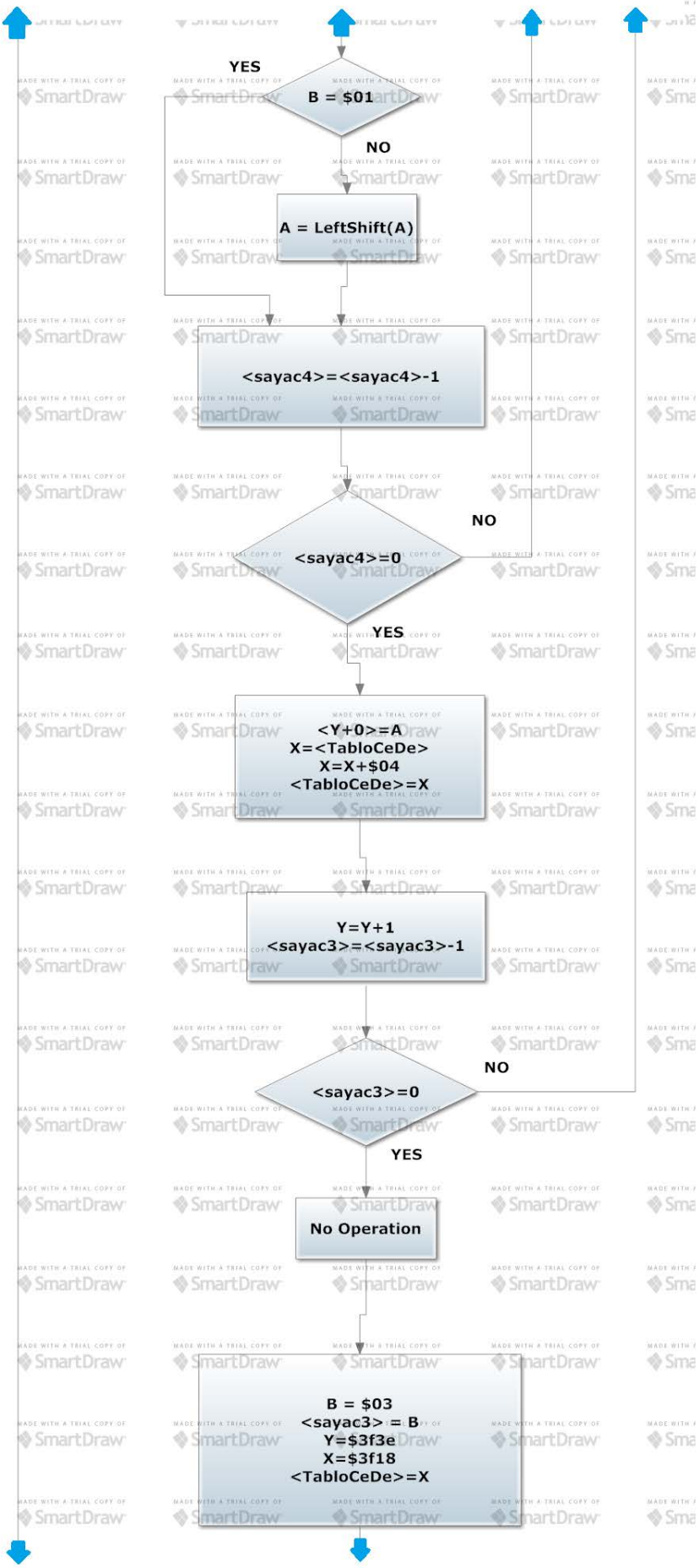
Şekil 4.2: Akış Şeması

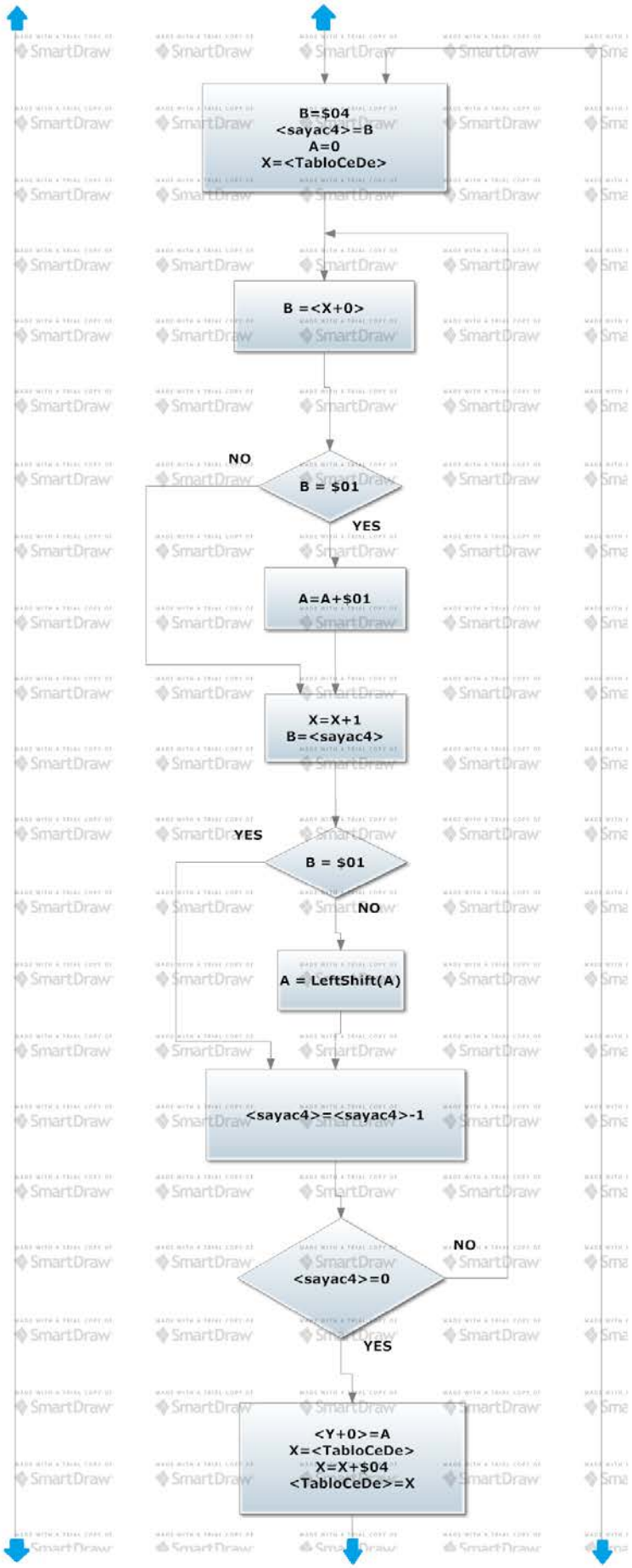

Clear Display komutunu LCD göstergeye gönder
Function set komutunu LCD göstergeye gönder
A=PORTE
A=(A) AND ($01)
A=$01
YES
NO
TripleDES_Programi
X = $4040
anaProgramKeyK_baslangicAdresi = X
X = $3ef8
anaProgramMesajM_baslangicAdresi = X
JSR
DES_Altprogrami
X = $5030
anaProgramKeyK_baslangicAdresi = X
X = $3ef8
anaProgramMesajM_baslangicAdresi = X
JSR
DES_DecryptionAltprogrami

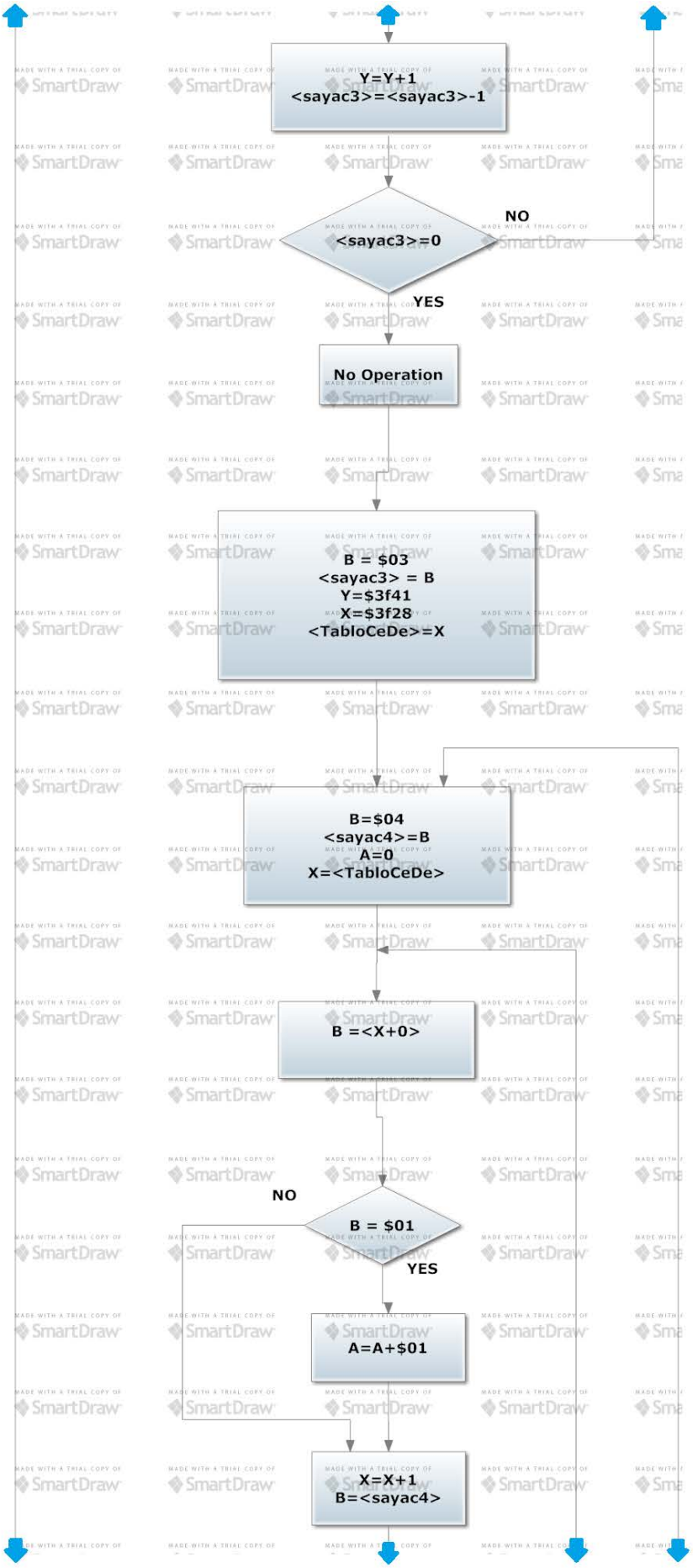

X = $4040
anaProgramKeyK_baslangicAdresi = X
X = $3ef8
anaProgramMesajM_baslangicAdresi = X
JSR
DES_Altprogrami
B = $03
<sayac3> = B
Y=$3f38
X=$3ef8
<TabloCeDe>=X
B=$04
<sayac4>=B
A=0
X=<TabloCeDe>
B =<X+0>

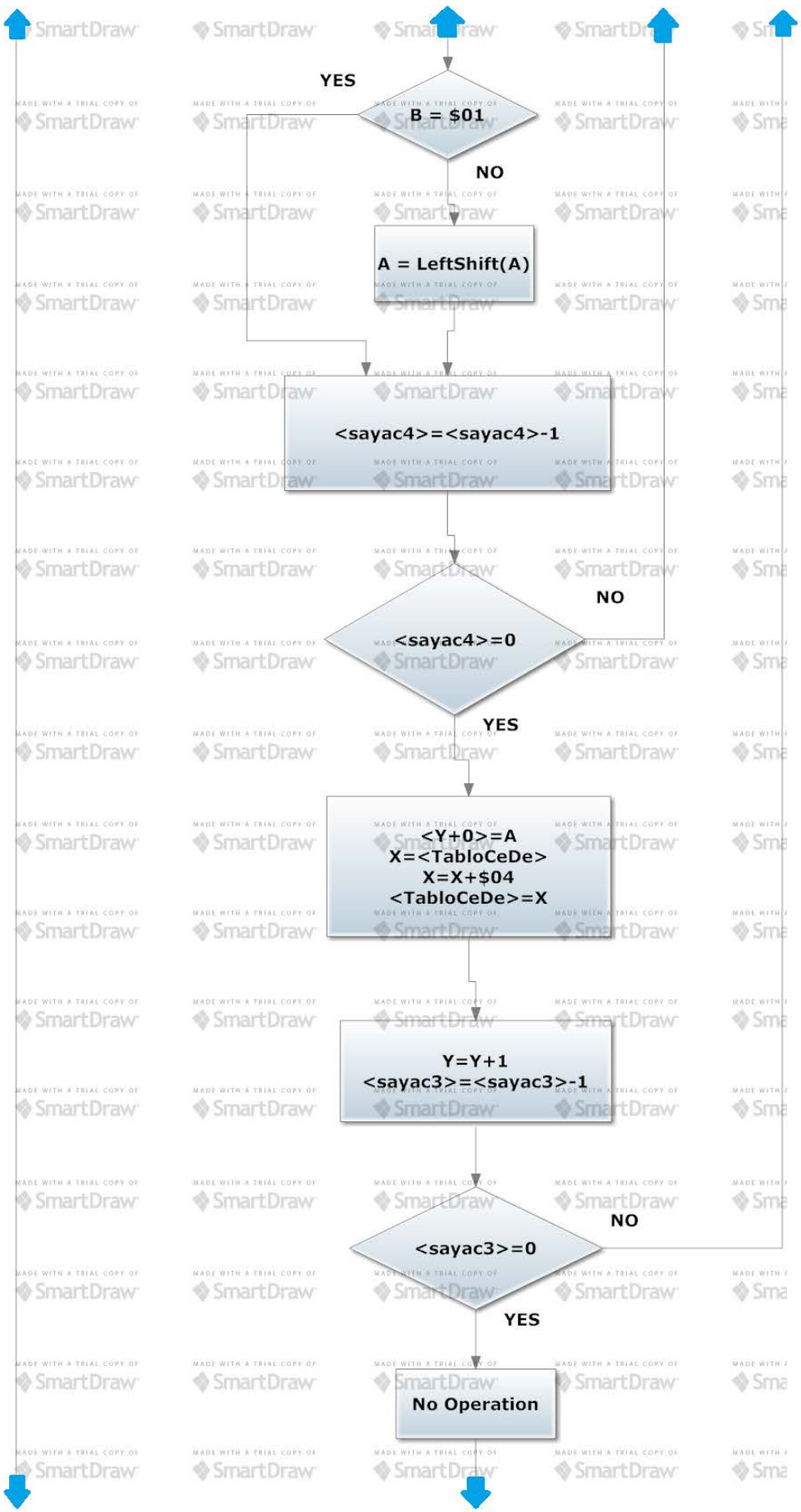

NO
B = $01
YES
A=A+$01
X=X+1
B=<sayac4>
YES
B = $01
NO
A = LeftShift(A)
<sayac4>=<sayac4>-1
NO
<sayac4>=0
YES
<Y+0>=A
X=<TabloCeDe>
X=X+$04
<TabloCeDe>=X

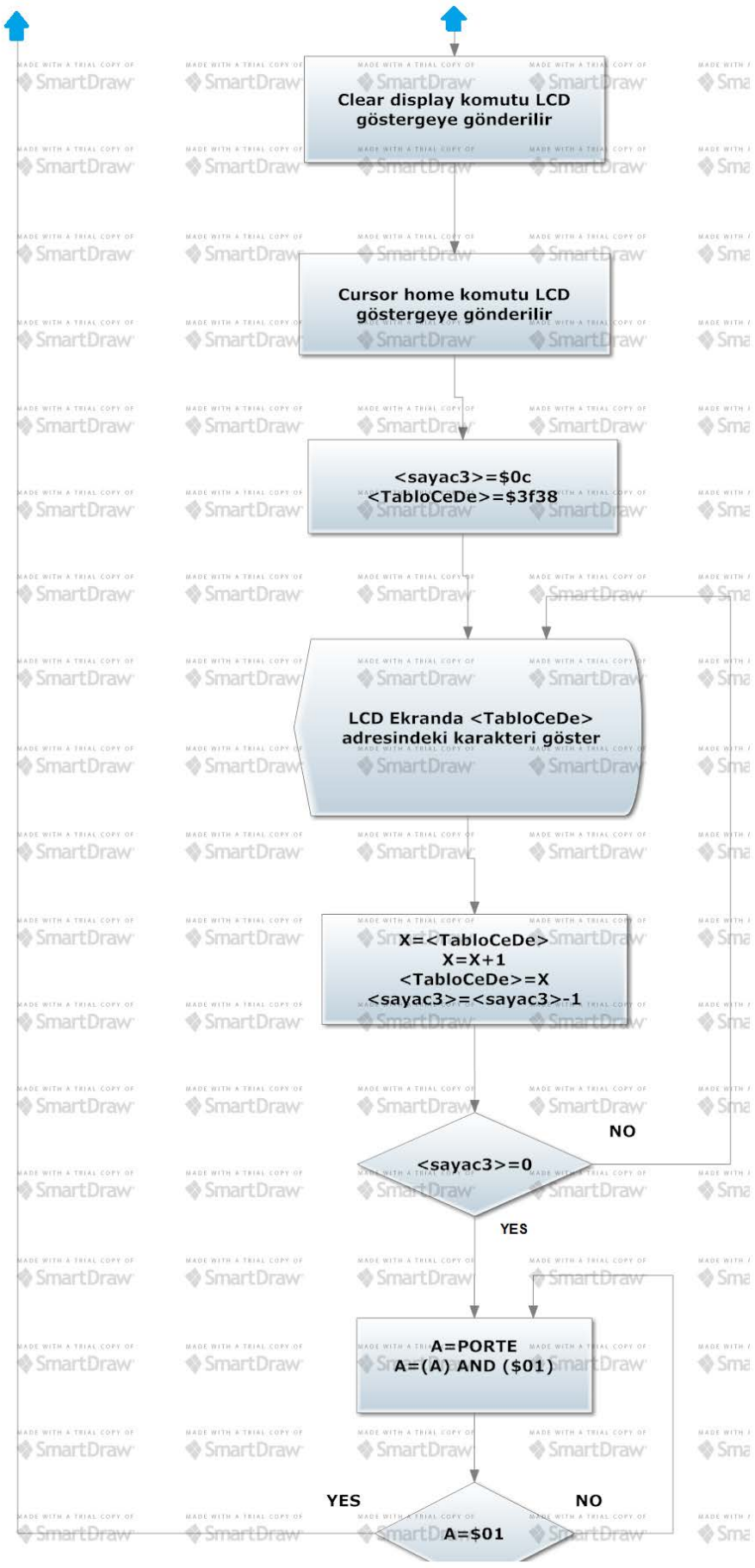

Y=Y+1
<sayac3>=<sayac3>-1
NO
<sayac3>=0
YES
No Operation
B = $03
<sayac3> = B
Y=$3f3b
X=$3f08
<TabloCeDe>=X
B=$04
<sayac4>=B
A=0
X=<TabloCeDe>
B =<X+0>
NO
B = $01
YES
A=A+$01
X=X+1
B=<sayac4>
SmartDraw


YES
B = $01
NO
A = LeftShift(A)
<sayac4>=<sayac4>-1
NO
<sayac4>=0
YES
<Y+0>=A
X=<TabloCeDe>
X=X+$04
<TabloCeDe>=X
Y=Y+1
<sayac3>=<sayac3>-1
NO
<sayac3>=0
YES
No Operation
B = $03
<sayac3> = B
Y=$3f3e
X=$3f18
<TabloCeDe>=X

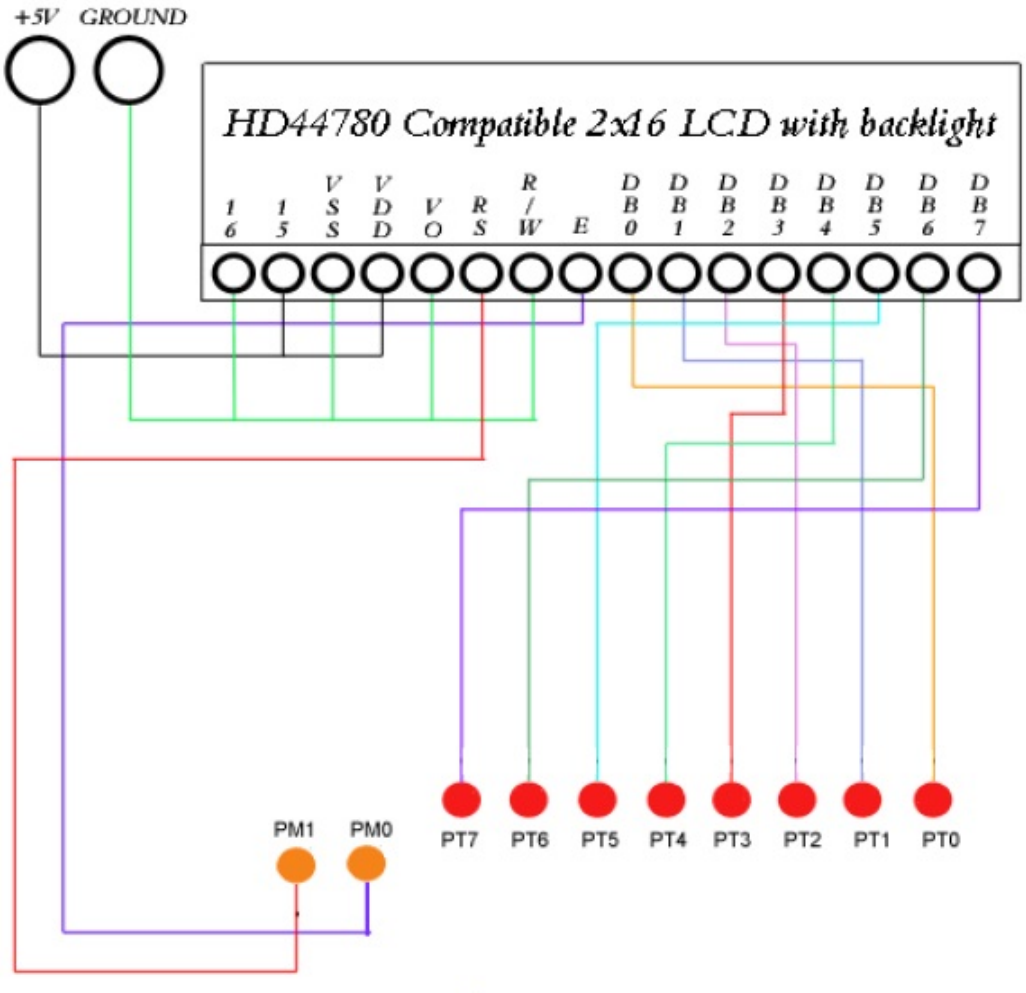

B=$04
<sayac4>=B
A=0
X=<TabloCeDe>
B =<X+0>
NO
B = $01
YES
A=A+$01
X=X+1
B=<sayac4>
YES
B = $01
NO
A = LeftShift(A)
<sayac4>=<sayac4>-1
NO
<sayac4>=0
YES
<Y+0>=A
X=<TabloCeDe>
X=X+$04
<TabloCeDe>=X


Y=Y+1
<sayac3>=<sayac3>-1
<sayac3>=0
NO
YES
No Operation
B = $03
<sayac3> = B
Y=$3f41
X=$3f28
<TabloCeDe>=X
B=$04
<sayac4>=B
A=0
X=<TabloCeDe>
B =<X+0>
NO
B = $01
YES
A=A+$01
X=X+1
B=<sayac4>
SmartDraw


YES
B = $01
NO
A = LeftShift(A)
<sayac4>=<sayac4>-1
NO
<sayac4>=0
YES
<Y+0>=A
X=<TabloCeDe>
X=X+$04
<TabloCeDe>=X
Y=Y+1
<sayac3>=<sayac3>-1
NO
<sayac3>=0
YES
No Operation


Clear display komutu LCD göstergeye gönderilir
Cursor home komutu LCD göstergeye gönderilir
<sayac3>=$0c
<TabloCeDe>=$3f38
LCD Ekranda <TabloCeDe> adresindeki karakteri göster
X=<TabloCeDe>
X=X+1
<TabloCeDe>=X
<sayac3>=<sayac3>-1
<sayac3>=0
NO
YES
A=PORTE
A=(A) AND ($01)
A=$01
YES
NO

Aşağıda projenin donanımı ile ilgili devre şeması verilmiştir. Projede MC9S12C32 mikrodenetçisini kullanan CSM12C32 geliştirme kiti kullanılmıştır. Projede LCD gösterge olarak HY-1602 LCD gösterge seçilmiştir. Bu projede LCD göstergenin veri yolu (DB7-DB0) CSM12C32 kitinin T PİAsına (PT7-PT0), LCD göstergenin izin girişi E kitin M PİAsının 0. bitine (PM0), LCD göstergenin kütük seçicisi RS kitin M PİAsının 1. bitine (PM1) bağlanmıştır.



Şekil 4.3: CSM12C32 ve HY-1602 Arasındaki Bağlantıların Şeması

5 TASARIM, GERÇEKLEŞTİRME VE TEST

Bu çalışmada tek kullanımlık parolalar üretebilen bir sistem tasarlanmıştır. Tek kullanımlık parolalar üretmek için DES algoritmasının özel bir hali olan 3-DES algoritması kullanılmıştır.

Projenin en önemli parçası şifreleme algoritmasıdır. Şifreleme algoritması mikroişlemci içine yerleştirilecek ve tek kullanımlık parolalar üretmek için kullanılacak kodları içerir. Bu kodlar bir ana program, bir şifreleme altprogramı ve bir de parola çözme altprogramından oluşur. Ana program içinde gerek duyulduğunda altprogramlar çağrılır ve böylece şifreleme veya parola çözme işlemleri gerçekleştirilir. 3-DES algoritması DES algoritmasının özel bir halidir. 3-DES algoritmasında iki adet 56 bitlik anahtar kullanılır. İlk anahtar verilen bir mesajı şifrelemek için kullanılır. İkinci anahtar ise şifrelenen mesajı çözmek için kullanılır. İkinci anahtar birinci anahtardan farklı olduğu için ikinci işlemde parola çözme işlemi yerine tekrar şifreleme işlemi gerçekleştirilmiş olur. Böylece verilen mesaj art arda iki kez şifrelenmiş olur. Son olarak mesaj birinci anahtar kullanılarak tekrar şifrelenir ve böylece başlangıçta elimizde bulunan mesaj art arda üç kez şifrelenmiş olur. 3-DES algoritması bu şekilde çalışır. Bu nedenle algoritma ana program, şifreleme altprogramı ve parola çözme altprogramından oluşacak şekilde tasarlanmıştır.

Bu çalışmada öncelikle DES algoritması örnek MİB üzerinde gerçekleştirilmiştir. Bu amaçla MikBil simülasyon paketi kullanılmıştır. Bu çalışmada 3-DES algoritması İTÜ Bilgisayar Mühendisliği bölümünde mikroişlemci derslerinde öğretilen örnek MİB üzerinde gerçekleştirilmiştir. Yazılım aynı mikroişlemci derslerinde kullanılan MikBil simulasyon paketi üzerinde yazılmıştır ve test edilmiştir. MikBil simulasyon paketi örnek MİB üzerinde kod yazılmasını ve yazılan kodların test edilmesini sağlayan bir platformdur [1]. Yapılan testlerde 3-DES algoritmasının verilen bir mesajı başarılı bir şekilde şifreleyebildiği gözlemlenmiştir.

5.1 Tek Kullanımlık Parola Üreten Sistem Yazılımı

Yazılım yukarıda söz edilen örnek MİB üzerinde gerçekleştirildikten sonra MC9S12C32 mikrodenetçisi üzerinde gerçekleştirilmiştir. Bu nedenle yazılım için bu mikrodenetçiyi kullanan CSM12C32 geliştirme kiti seçilmiştir. Bu bölümde tek kullanımlık parola üreten sistem ile ilgili kodların açıklamaları verilmiştir.

```
Entry:
_Startup:

            ; remap the RAM &amp;amp; EEPROM here. See EB386.pdf
 ifdef _HCS12_SERIALMON
            ; set registers at $0000
            CLR   $11                   ; INITRG= $0
            ; set ram to end at $3FFF
            LDAB  #$39
            STAB  $10                   ; INITRM= $39

            ; set eeprom to end at $0FFF
            LDAA  #$9
            STAA  $12                   ; INITEE= $9


            LDS   #$3FFF+1          ; See EB386.pdf, initialize the stack pointer
 else
            LDS   #RAMEnd+1         ; initialize the stack pointer
 endif



            ;LDS   #RAMEnd+1         ; initialize the stack pointer

            CLI                      ; enable interrupts
```

Şekil 5.1: Kod Açıklamaları 1

Yukarıda Şekil 5.1'de yazılımın giriş bölümü gösterilmiştir. Bu bölümde yığın göstergesine ilk değer yüklenmektedir ve bazı sistem ayarlamaları yapılmaktadır.

```
JSR   ilkIslemler   ;T ve M portlari cikis yapiliyor

LDAB  #$01     ;E = 1 yapiliyor
JSR   egonder
LDAA  #$0e      ;#$0f     ;Display on_off control komutu gonderiliyor
LDAB  #$01
JSR   gonder
LDAB  #$00     ;E = 0 yapilarak LCDnin komutu
JSR   egonder ;islemesi saglaniyor
JSR   delayMS ;LCDnin komutu islemesi icin bekleniyor


LDAB  #$01
JSR   egonder
LDAA  #$01     ;Clear display komutu gonderiliyor
LDAB  #$01
JSR   gonder
LDAB  #$00
JSR   egonder
JSR   delayMS


LDAB  #$01
JSR   egonder
LDAA  #$38     ;Function set komutu gonderiliyor
LDAB  #$01
JSR   gonder
LDAB  #$00
JSR   egonder
JSR   delayMS

;CLI    ; enable interrupts

;Bu komutlardan sonra artık LCD karakter gostermeye hazir hale getirildi
```

Şekil 5.2: Kod Açıklamaları 2

Burada kod içinde LCD göstergenin çalışması için başlangıç ayarları yapılmıştır. Öncelikle LCD göstergeye Display on_off control komutu gönderilmiştir. Burada göstergenin açılması sağlanmıştır ve imleç açık olarak ayarlanmıştır. İmlecin yanıp sönmesi ayarı kapalı olarak bırakılmıştır. Gösterge açma ayarı yapıldıktan sonra göstergeye Clear display komutu gönderilmiştir. Bu komut ekranı temizler ve imleci başlangıç konumuna getirir. Daha sonra göstergeye Function set komutu gönderilmiştir. Bu komut arayüz veri uzunluğu, karakter boyutu ve bir satırdaki karakter sayısı ile ilgili ayarlamaları yapar.

```
dongu1_AnaProgramDongusu:
            ;SEI            ; Disable interrupts

            ;CLI          ; enable interrupts

            ;JSR  delay1sec

            ;JSR  delay1sec

            LDAA PORTE ; read SW1 at PORTE0
;SW1 dugmesine basildiysa PORTE bit0 sifir degerini alir ama dugmeye basilmadiysa 1 degerinde kalir
            ANDA #$01
            CMPA #$01

            ;CMPA #$01
            BEQ  dongu1_AnaProgramDongusu

            BRA  TripleDES_Programi
```

Şekil 5.3: Kod Açıklamaları 3

Burada Şekil 5.3'de tek kullanımlık parola üreten sistemin ana program döngüsü gösterilmiştir. Bu projede yapılan tasarıma göre CSM12C32 geliştirme kiti üzerindeki SW1 düğmesine her basıldığında bir tek kullanımlık parola üretilir. Yazılımda ana program döngüsü içinde SW1 düğmesine basılıp basılmadığı kontrol edilir. CSM12C32 geliştirme kitinde SW1 düğmesi MC9S12C32 mikrodenetçisinin PORTE iskelesinin 0. bitine bağlanmıştır. SW1 düğmesine basıldıysa PORTE bit0 sıfır değerini alır; ama düğmeye basılmadıysa PORTE bit0 lojik-1 değerinde kalır. Burada SW1 düğmesine basılıp basılmadığı kontrol ediliyor. Öncelikle PORTE iskelesinin içeriği A akümülatörüne yükleniyor. Daha sonra A akümülatörü $01 değeri ile VE işlemine giriyor. Bunun nedeni burada PORTE iskelesinin sadece 0. biti ile ilgilenilmesidir. Daha sonra A akümülatörünün içeriği $01 değeri ile karşılaştırılıyor. Eğer PORTE bit0 lojik-1 değerinde kaldıysa dongu1_AnaProgramDongusu etiketli ana program döngüsü başına dallanılır. Eğer PORTE bit0 lojik-0 değerini aldıysa TripleDES_Programi etiketli 3-DES şifreleme programına dallanılır.

```
gonder:
            STAB PTM   ;B akumulator degeri M portuna verilir
            STAA PTT   ;A akumulator degeri T portuna verilir
            JSR  delayMS  ;gerekli bekleme saglanir
            RTS

egonder:
            STAB PTM   ;B akumulator degeri M portuna verilir
            JSR  delayMS  ;gerekli bekleme saglanir
            RTS


ilkIslemler:          ;M ve T portlari cikis olarak belirlenir
            LDAA #$FF
            STAA DDRT
            STAA DDRM
            RTS

delay1sec:
            LDY  #$0fff    ;long delay by
dly1Loop:
            JSR  delayMS   ;Y*delayMS
            DEY
            BNE  dly1Loop
            RTS


delayMS:
            LDX  #$02ff    ;#$00ff    ;short delay
dlyMSLoop:
            NOP            ;X*NOP
            DEX
            BNE  dlyMSLoop
            RTS
```

Şekil 5.4: Kod Açıklamaları 4

Burada Şekil 5.4'de LCD gösterge ile ilgili alt programlar vardır. LCD gösterge kendisine gönderilen komutları E izin girişi sinyalinin düşen kenarında değerlendirir. LCD göstergeye her komut gönderildiğinde LCD göstergenin komutları işlemesi için belli bir zaman geçer. Bu nedenle belli bir değerde bekleme sağlayan bir alt programın, LCD göstergeye her komut gönderildikten sonra kullanılması gereklidir. Burada delayMS alt programı bu beklemeyi sağlar. Bu projede LCD göstergenin veri yolu (DB7-DB0) CSM12C32 kitinin T PİAsına (PT7-PT0), LCD göstergenin izin girişi E kitin M PİAsının 0. bitine (PM0), LCD göstergenin kütük seçicisi RS kitin M PİAsının 1. bitine (PM1) bağlanmıştır. Bu nedenle kitin M ve T iskeleleri çıkış olarak ayarlanmalıdır. Burada ilkIslemler alt programı bu işlemleri yapar. Burada gonder alt programı LCD göstergeye komut göndermek için kullanılır. Burada egonder alt programı ise LCD göstergeye izin girişi (E) sinyali göndermek için kullanılır.

```
TripleDES_Programi:


;*K ile gosterilen 64 bitlik Key bellekte $4040-$407F arasinda bulunmaktadir*
;*M olarak verilen mesaj bellekte $4810 ve $484F adresleri arasında bulunur($484F adresinde son elemar
;*KeyK2 tablosu bellekte $5030 ve $506f adresleri arasinda yer alir

;*************************************************************************
;* Sifrelenmis mesaj SifrelenmisC olarak gosterilir ve bu veri
;* bellekte $0ef8 ve $0f37 adresleri arasinda bulunur($0f37 adresinde son eleman vardir)
;*************************************************************************


            LDX #$4040
            STX anaProgramKeyK_baslangicAdresi
            ;LDX #$4810
            LDX #$3ef8
            STX anaProgramMesajM_baslangicAdresi

;******DES altprogramina dallan
            JSR DES_Altprogrami

;DES_Altprogrami
;DES_DecryptionAltprogrami


            LDX #$5030
            STX anaProgramKeyK_baslangicAdresi
            LDX #$3ef8
            STX anaProgramMesajM_baslangicAdresi

;******DES Decryption altprogramina dallan

            JSR DES_DecryptionAltprogrami



            LDX #$4040
            STX anaProgramKeyK_baslangicAdresi
            LDX #$3ef8
            STX anaProgramMesajM_baslangicAdresi

;******DES altprogramina dallan
            JSR DES_Altprogrami
```

Şekil 5.5: Kod Açıklamaları 5

Burada Şekil 5.5'de TripleDES_Programi alt programının başlangıç bölümü gösterilmiştir. 3-DES algoritması DES algoritmasının özel bir halidir. 3-DES algoritmasında iki adet 56 bitlik anahtar kullanılır. İlk anahtar verilen bir mesajı şifrelemek için kullanılır. İkinci anahtar ise şifrelenen mesajı çözmek için kullanılır. İkinci anahtar birinci anahtardan farklı olduğu için ikinci işlemde parola çözme işlemi yerine tekrar şifreleme işlemi gerçekleştirilmiş olur. Böylece verilen mesaj art arda iki kez şifrelenmiş olur. Son olarak mesaj birinci anahtar

kullanılarak tekrar şifrelenir ve böylece başlangıçta elimizde bulunan mesaj art arda üç kez şifrelenmiş olur. 3-DES algoritması bu şekilde çalışır. Bu nedenle algoritma ana program, şifreleme altprogramı ve parola çözme altprogramından oluşacak şekilde tasarlanmıştır. Burada öncelikle şifreleme işlemi için kullanılan ilk anahtarın bellekteki adresi anaProgramKeyK_baslangicAdresi isimli değişkene yazılmaktadır. Daha sonra 3-DES algoritmasının oluşturacağı parolaları yazacağı adres aralığının başlangıç değeri olan **$3ef8** değeri anaProgramMesajM_baslangicAdresi isimli değişkene yazılır. Daha sonra bu değişkenler üzerinde işlem yapacak olan DES_Altprogramı alt programına dallanılır. Daha sonra bu işlemlere benzer şekilde ikinci anahtar değeri de ilgili değişkene yazılır ve DES şifre çözme alt programına dallanılır. DES şifre çözme altprogramına birinci anahtardan farklı olan ikinci anahtar ile birlikte dallanıldığı için bu noktada daha önce üretilmiş parola değeri tekrar şifreleme işlemine girer. Daha sonra birinci anahtar ile tekrar DES şifreleme alt programı çağrılır ve böylece başlangıçta elimizde bulanan mesaj değeri art arda üç kez şifrelenmiş olur. 3-DES algoritması bu şekilde çalıştığı için yazılım bu şekilde tasarlanmıştır.

```
;*************************************************************************
;*     Simdi 16 haneli SifrelenmisC mesajini 12 haneli SifrelenmisHexadecimalResult     *
;*     mesajina cevirириz                                                                *
;*************************************************************************


;SifrelenmisHexadecimalSonuc bellekte $0f38 ve $0f43 adresleri arasinda yer alir*

;*************************************************************************
;*  Sifrelenmis mesaj SifrelenmisC olarak gosterilir ve bu veri
;*  bellekte $0ef8 ve $0f37 adresleri arasinda bulunur($0f37 adresinde son eleman vardir)
;*************************************************************************


;*************************  Parca 1  *******************************
        LDAB #$03
        STAB sayac3
        LDY  #$3f38

        LDX  #$3ef8
        STX  TabloCeDe

dongu1_TripleDES_parca1:
        LDAB #$04
        STAB sayac4
        CLRA
        LDX  TabloCeDe

geri1_TripleDES_parca1:
        LDAB 0,X
        CMPB #$01
        BNE  A_yaDegerEkledim_TripleDES_parca1
        ADDA #$01
A_yaDegerEkledim_TripleDES_parca1:
        INX
        LDAB sayac4
        CMPB #$01
        BEQ  DidNotShiftA_parca1
        LSLA

DidNotShiftA_parca1:
        DEC  sayac4
```

```
        BNE  geri1_TripleDES_parca1
devam1_TripleDES_parca1:
        STAA 0,Y
        LDX  TabloCeDe
        LDAB #$04
        ABX
        STX  TabloCeDe

        INY
        DEC  sayac3
        BNE  dongu1_TripleDES_parca1
devam2_TripleDES_parca1:
        NOP




;************************  Parca 2  *******************************
        LDAB #$03
        STAB sayac3
        LDY  #$3f3b

        LDX  #$3f08
        STX  TabloCeDe

dongu1_TripleDES_parca2:
        LDAB #$04
        STAB sayac4
        CLRA
        LDX  TabloCeDe

geri1_TripleDES_parca2:
        LDAB 0,X
        CMPB #$01
        BNE  A_yaDegerEkledim_TripleDES_parca2
        ADDA #$01
A_yaDegerEkledim_TripleDES_parca2:
        INX
        LDAB sayac4
        CMPB #$01
        BEQ  DidNotShiftA_parca2
        LSLA

DidNotShiftA_parca2:
        DEC  sayac4
        BNE  geri1_TripleDES_parca2
devam1_TripleDES_parca2:
        STAA 0,Y
```

```
        LDX  TabloCeDe
        LDAB #$04
        ABX
        STX  TabloCeDe

        INY
        DEC  sayac3
        BNE  dongu1_TripleDES_parca2
devam2_TripleDES_parca2:
        NOP




;************************  Parca 3  *******************************
        LDAB #$03
        STAB sayac3
        LDY  #$3f3e

        LDX  #$3f18
        STX  TabloCeDe

dongu1_TripleDES_parca3:
        LDAB #$04
        STAB sayac4
        CLRA
        LDX  TabloCeDe

geri1_TripleDES_parca3:
        LDAB 0,X
        CMPB #$01
        BNE  A_yaDegerEkledim_TripleDES_parca3
        ADDA #$01
A_yaDegerEkledim_TripleDES_parca3:
        INX
        LDAB sayac4
        CMPB #$01
        BEQ  DidNotShiftA_parca3
        LSLA

DidNotShiftA_parca3:
        DEC  sayac4
        BNE  geri1_TripleDES_parca3
devam1_TripleDES_parca3:
        STAA 0,Y
        LDX  TabloCeDe
        LDAB #$04
```

```
        ABX
        STX  TabloCeDe

        INY
        DEC  sayac3
        BNE  dongu1_TripleDES_parca3
devam2_TripleDES_parca3:
        NOP



;************************  Parca 4  *******************************
        LDAB #$03
        STAB sayac3
        LDY  #$3f41

        LDX  #$3f28
        STX  TabloCeDe

dongu1_TripleDES_parca4:
        LDAB #$04
        STAB sayac4
        CLRA
        LDX  TabloCeDe

geri1_TripleDES_parca4:
        LDAB 0,X
        CMPB #$01
        BNE  A_yaDegerEkledim_TripleDES_parca4
        ADDA #$01
A_yaDegerEkledim_TripleDES_parca4:
        INX
        LDAB sayac4
        CMPB #$01
        BEQ  DidNotShiftA_parca4
        LSLA

DidNotShiftA_parca4:
        DEC  sayac4
        BNE  geri1_TripleDES_parca4
devam1_TripleDES_parca4:
        STAA 0,Y
        LDX  TabloCeDe
        LDAB #$04
        ABX
        STX  TabloCeDe

        INY
        DEC  sayac3
```

```
        BNE  dongu1_TripleDES_parca4
devam2_TripleDES_parca4:
        NOP
```

Tablo 5.1: Kod Açıklamaları

Burada Tablo 5.1'de tek kullanımlık parola üreten sistemin kullandığı Hash algoritmasının kodları gösterilmiştir. Burada 3-DES algoritmasının ürettiği 16 haneli parola Hash fonksiyonuna girer ve sonuç olarak 12 haneli SifrelenmisHexadecimalSonuc parolası oluşturulur. Bu 12 haneli parola daha sonra LCD göstergede gösterilir. Hash fonksiyonu güvenliğin arttırılması için gereklidir. Hash fonksiyonu tek kullanımlık parolalar üretmek için kullanılan 3-DES algoritmasının kullandığı anahtarları ve son üretilen tek kullanımlık parolayı ele geçiren birinin bir sonraki üretilecek parolayı bilmesini engeller. Bu projede basit bir Hash fonksiyonu tasarlanmıştır. Bu Hash fonksiyonu 3-DES algoritması ile üretilen 16 haneli parolanın içinden 12 haneyi seçer ve geri kalan 4 haneyi kullanmaz. Üretilen tek kullanımlık parolanın ilk 3 hanesi seçilir ve 4. hanesi kullanılmaz. Benzer şekilde 5,6 ve 7. haneler de Hash algoritması tarafından seçilir ve 8. hane kullanılmaz. Daha sonra Hash algoritması 9,10 ve 11. haneleri de seçer ve 12. Hane kullanılmaz ve son olarak 13,14 ve 15. haneler seçilir ve 16. hane kullanılmaz. Böylece tasarlanan Hash fonksiyonu 3-DES algoritmasının ürettiği 16 hanelik parola içinden 1, 2, 3, 5, 6, 7, 9, 10, 11, 13, 14 ve 15. haneleri üretilecek 12 hanelik sonuç parolası için seçer ve geri kalan haneleri de kullanmaz. Projede tasarlanan Hash algoritması bu şekilde çalışır.

```
;Simdi sonuclari LCD ekranda gosterelim

        LDAB #$01
        JSR  egonder
        LDAA #$01   ;Clear display komutu gonderiliyor
        LDAB #$01
        JSR  gonder
        LDAB #$00
        JSR  egonder
        JSR  delayMS


        LDAB #$01
        JSR  egonder
        LDAA #$02   ;#$03    ;Cursor home komutu gonderiliyor
        LDAB #$01
        JSR  gonder
        LDAB #$00
        JSR  egonder
```

```
        JSR  delayMS



        LDAB #$0c
        STAB sayac3

;SifrelenmisHexadecimalSonuc bellekte $0f38 ve $0f43 adresleri arasinda yer alir*
        LDX  #$3f38
        STX  TabloCeDe

geri1_AnaProgramDongusu:
        LDAB #$01
        JSR  egonder

        LDX  TabloCeDe
        LDAA 0,X

        CMPA #$0A
        BHS  DegerDahaBuyuk_AnaProgramDongusu
        ADDA #$30
        BRA  Devam1_AnaProgramDongusu
DegerDahaBuyuk_AnaProgramDongusu:
        TAB       ;Transfer A to B
        SUBB #$0A
        TBA       ;Transfer B to A
        ADDA #$41

Devam1_AnaProgramDongusu:
        LDAB #$03
        JSR  gonder

        LDAB #$00
        JSR  egonder
        JSR  delayMS

        LDX  TabloCeDe
        INX
        STX  TabloCeDe

        DEC  sayac3
        BNE  geri1_AnaProgramDongusu

ileri1_AnaProgramDongusu:
        NOP



TripleDES_ProgramiDugmeDongusuSon:
        LDAA PORTE ; read SW1 at PORTE0
```

```
;SW1 dugmesine basildiysa PORTE bit0 sifir degerini alir ama dugmeye basilmadiysa 1 degerinde kalir
        ANDA #$01
        CMPA #$01

        ;CMPA #$01

        BNE  TripleDES_ProgramiDugmeDongusuSon
        LBRA  dongu1_AnaProgramDongusu
```

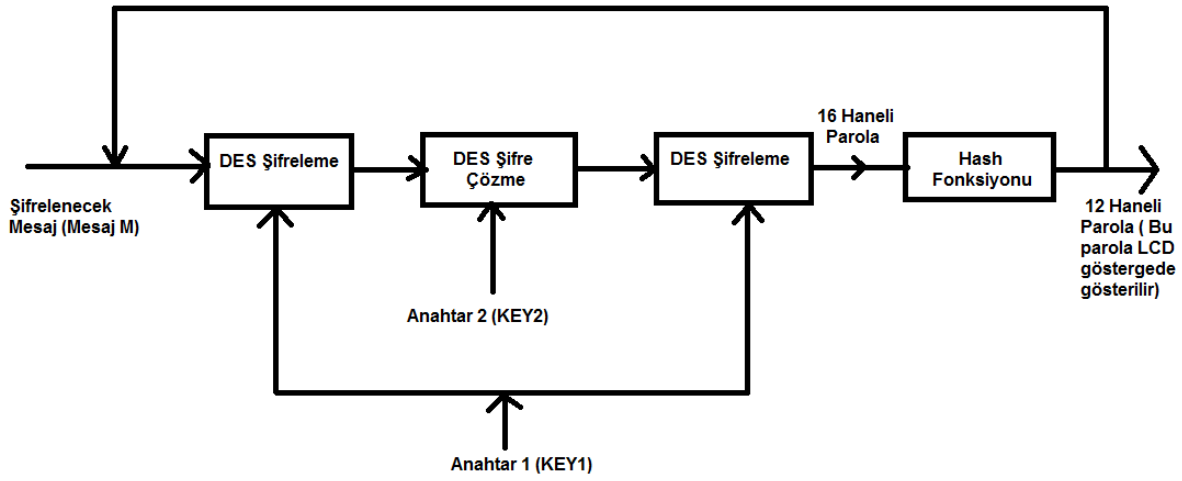
Tablo 5.2: Kod Açıklamaları

Yazılımın bu bölümünde tek kullanımlık parolalar LCD göstergede gösterilmektedir. Öncelikle LCD göstergeye Clear Display komutu gönderilir ve LCD göstergenin ekranı temizlenir. Daha sonra göstergeye Cursor Home komutu gönderilir ve imleç başlangıç pozisyonuna alınır. Bu yazılımda SifrelenmisHexadecimalSonuc adresinden itibaren belleğe yazılan tek kullanımlık parola bellekte $3f38 ve $3f43 adresleri arasında yer alır. Bu parola 12 hanelidir. Bu nedenle öncelikle sayac3 değişkenine $0c değeri yazılır. Daha sonra parolanın bellekteki başlangıç adresi olan $3f38 değeri TabloCeDe değişkenine yazılır. Daha sonra geri1_AnaProgramDongusu döngüsüne girilir. Bu döngünün her turunda öncelikle E izin girişi 1 yapılır ve daha sonra parolanın bir karakteri $0A değeri ile karşılaştırılır. Eğer parola karakteri 16 tabanındaki sayı sisteminde $0 ve $9 değerleri arasında ise bu karakter değerine $30 eklenir ve oluşan sonuç veri olarak LCD göstergeye gönderilir ve daha sonra E izin girişi 0 yapılarak LCD göstergenin karakteri ekranda göstermesi sağlanır. Burada karakter değerine $30 eklenmesinin nedeni ASCII karakter kodlamasına göre 0 karakterinin 16 sayı tabanında $30 ile temsil edilmesidir. Benzer şekilde ASCII kodlamasına göre 9 değeri 16 sayı tabanında $39 ile temsil edilir. Eğer parola karakteri $0A değerinden büyükse veya bu değere eşitse bu durum parola karakterinin $A ve $F değerleri arasında olduğunu gösterir. Parola karakterinin LCD göstergede gösterilebilmesi için bu karakteri ASCII karakter kodlamasına göre temsil eden değer LCD göstergeye gönderilmelidir. ASCII karakter kodlamasına göre A karakteri $41 değeri ile temsil edilmektedir. Benzer şekilde ASCII kodlamasına göre F karakteri $F karakteri ile temsil edilmektedir. Bu nedenle parola karakterinden öncelikle $0A değeri çıkartılır ve daha sonra oluşan sonuç değerine $41 değeri eklenir. Böylece parola karakterini ASCII kodlamasına göre temsil eden değer A akümülatöründe elde edilir. Daha sonra A akümülatörünün içeriği veri olarak LCD göstergeye gönderilir. Daha sonra E izin girişi 0 yapılarak LCD göstergenin parola karakterini ekranda göstermesi sağlanır. Daha sonra döngünün son adımında TabloCeDe değişkeninin değeri bir arttırılır. Böylece bir sonraki döngü turunda bir sonraki parola karakteri LCD ekranda gösterilecektir. Bu döngü 12 tur çalışır ve böylece şifreleme ve Hash algoritmasının ürettiği 12 hanelik parola LCD göstergede gösterilir.

Yazılımda ana programın sonunda TripleDES_ProgramiDugmeDongusuSon etiketli küçük bir döngüye gelinir. Burada SW1 düğmesinin basılı olup olmadığına bakılır. Eğer SW1 düğmesi basılı ise bu döngü içinde kalınır. Eğer SW1 düğmesi basılı değilse dongu1_AnaProgramDongusu etiketli ana program döngüsüne geri dönülür. Daha önce bahsedildiği gibi tek kullanımlık parolalar üretmek için CSM12C32 geliştirme kitindeki SW1 düğmesi kullanılır. SW1 düğmesine basıldığında ana program döngüsünde

TripleDES_Programi etiketli bölgeye dallanılıyor. Burada tek kullanımlık bir parola üretiliyor ve bu parola Hash fonksiyonuna girdikten sonra LCD göstergede gösteriliyor. En sonunda program TripleDES_ProgramiDugmeDongusuSon etiketli adrese geliyor. Program bu noktaya geldiğinde SW1 düğmesine hala basılıysa, program TripleDES_ProgramiDugmeDongusuSon etiketli bu düğme döngüsünde kalıyor ve ana programa dönülmüyor. Eğer SW1 kullanıcı düğmesinin basılı konumu bırakılırsa bu durumda program bu düğme döngüsünden çıkıyor ve ana programa dönüyor. Ana programın sonundaki bu düğme döngüsü SW1 düğmesine bir kere basıldığında birden fazla parola üretilmemesi için gereklidir. SW1 kullanıcı düğmesindeki titreşimler de dikkate alındığında ana program sonundaki bu düğme döngüsünün gerekli olduğu görülür.

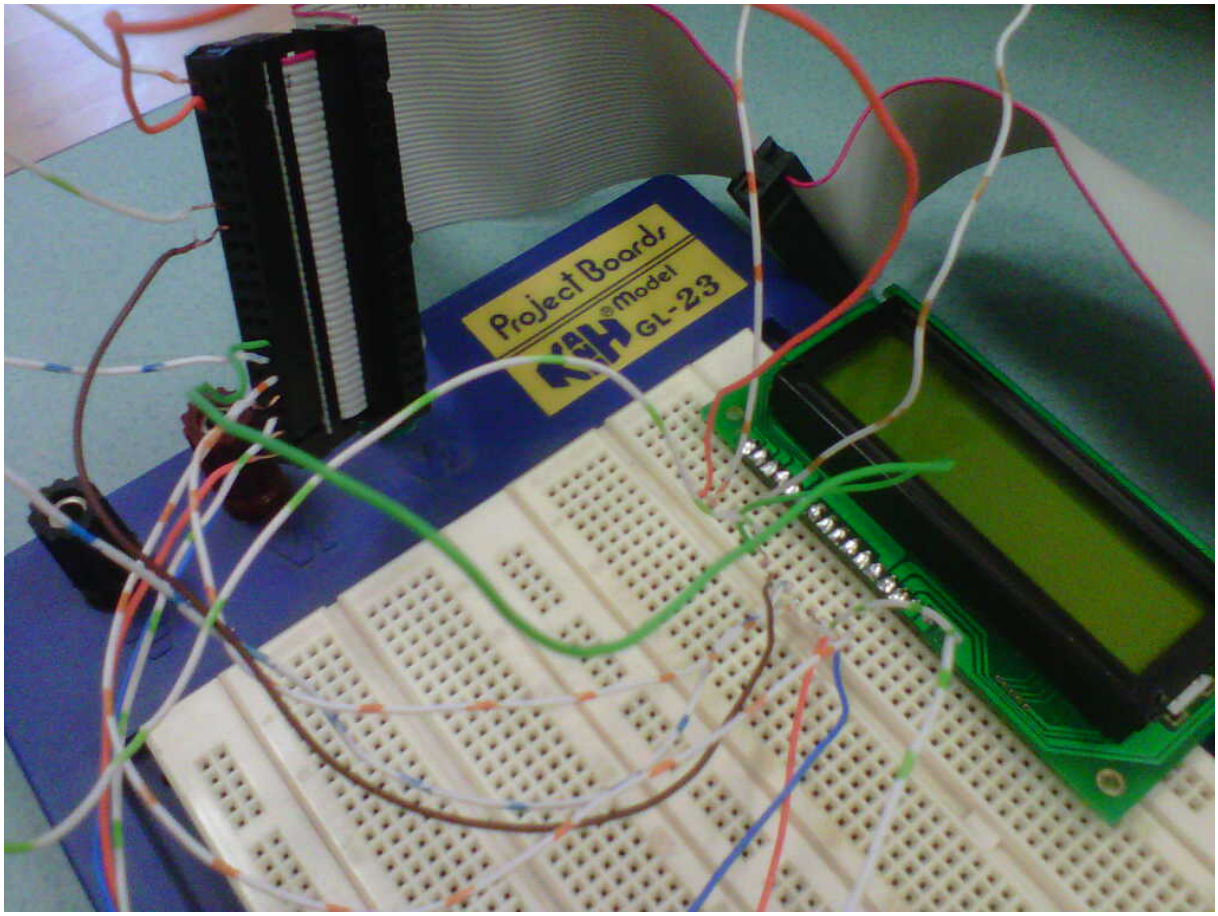
Projede tasarlanan tek kullanımlık parola sisteminin kullanıcı tarafı ile ilgili olan kodlar bunlardır. Burada ana program açıklanmıştır. Bu ana program dışında bir DES_Altprogrami ve bir de DES_DecryptionAltprogrami alt programları vardır. Bu alt programların burada açıklanmasına gerek yoktur, çünkü bu altprogramlar içinde yeterli derecede yorum satırları vardır. Aşağıda Şekil 5.6'da tek kullanımlık parola sisteminin genel şeması görülmektedir. Bu şemada tek kullanımlık parola sisteminin nasıl çalıştığı gösterilmiştir.



Şekil 5.6: Sistemin Genel Şeması

6 DENEYSEL SONUÇLAR

Bu projede tek kullanımlık parola üreten bir sistem tasarlanmıştır. Tek kullanımlık parolalar üretmek için 3-DES algoritması kullanılmıştır. Projede MC9S12C32 mikrodenetçisini kullanan CSM12C32 geliştirme kiti üzerinde çalışılmıştır. Kullanıcı CSM12C32 kiti üzerindeki SW1 düğmesine her bastığında bir tek kullanımlık parola üretilir ve bu parola HY-1602 LCD göstergede gösterilir. Projede yazılan kodun deneme aşamasında Code Warrior Development Studio for S12(X) IDE ortamında yazılan kod RS232 Serial bağlantı üzerinden CSM12C32 geliştirme kitine yüklenmiştir. CSM12C32 geliştirme kiti ile LCD gösterge arasındaki bağlantıyı 40 damarlı ara bağlantı kablosu sağlar.



Şekil 6.1: Başlangıçta Gerçeklenen Deney Düzeneği

Burada Şekil 6.1'de genişletilmiş deney düzeneği görülüyor. Başlangıçta yazılan kod bu düzenek üzerinde test edilmiştir ve kodun doğru bir şekilde tek kullanımlık parolalar ürettiği görülmüştür. Daha sonra CSM12C32 geliştirme kiti ve HY-1602 LCD gösterge arasındaki

bağlantılar, 40 damarlı ara bağlantı kablosu üzerinde çeşitli kesme ve lehimleme işlemleri yapılarak kurulmuştur. Böylece CADET eğitim kitine ihtiyaç olmadan deney düzeneğini gerçekleme olanağı doğmuştur. Böylece taşınabilir bir deney düzeneği elde edilmiştir.

Şekil 6.2: CSM12C32 kiti ve Ara Bağlantı Kablosu

Deneyde tek kullanımlık parola üreten yazılım CSM12C32 geliştirme kitine yüklendikten sonra geliştirme kiti üzerindeki RESET düğmesine basılmıştır. Bu anda LCD göstergede imleç başlangıç pozisyonunda görülmüştür. Daha sonra kit üzerindeki SW1 kullanıcı düğmesine her basıldığında bir tek kullanımlık parola üretildiği ve bu parolanın LCD göstergede gösterildiği görülmüştür.

Şekil 6.3: Tek Kullanımlık Parolaların LCD Göstergede Gösterilişi

Kit üzerindeki SW1 düğmesine ikinci kez basıldığında yeni bir tek kullanımlık parola üretilir ve bu parola LCD göstergede gösterilir.

Şekil 6.4: İkinci Turda Üretilen Tek Kullanımlık Parola

Bu şekilde SW1 kullanıcı düğmesine her basıldığında yeni bir tek kullanımlık parola üretilir ve bu parola LCD göstergede gösterilir.

Şekil 6.5: Beşinci Turda Üretilen Tek Kullanımlık Parola

Böylece gerçekleştirilen deneylerde projede tasarlanan yazılım ve donanımın başarılı bir şekilde tek kullanımlık parolalar üretebildiği ve bu parolaları LCD göstergede gösterebildiği görülmüştür.

7 SONUÇ VE ÖNERİLER

Bu bölümde projede gerçekleştirilen tek kullanımlık parola sistemi fiyat, performans ve çevre faktörleri açısından ele alınacaktır. Bu projede tek kullanımlık parola üreten bir sistem gerçekleştirilmiştir.

OTP adı verilen tek kullanımlık parola üreten sistemler güvenliği arttırmak amacıyla tasarlanmışlardır. OTP sistemlerinde tek bir bağlanma işlemi veya hesap işlemi için bir parola üretilir ve bu parola her işlemde değişir. Parola her bir işlem için tekrar üretilir. Bu durum daha önce hesap işlemi yapmak için kullanılmış bir parolayı bir şekilde elde eden kötü niyetli kimselerin güvenliği aşarak hesaba giriş yapabilmesini olanaksız kılar, çünkü daha önce hesaba giriş yapmak için kullanılmış olan parola artık geçersizdir ve değişmiştir. Bu tek kullanımlık parola üreten sistemlerin güvenlik konusunda önemli bir performans gösterdiğini ortaya koyuyor. OTP sistemleri genellikle diğer güvenlik sistemleri ile birlikte kullanılır ve katmanlı bir güvenlik sisteminin güçlü bir katmanını oluştururlar. Örneğin OTP parolalar statik parolalar ile birlikte kullanılabilir. Kullanıcı hesap işlemi yapmak için sisteme bağlanırken öncelikle statik bir parola ile sisteme girebilir ve ardından sistem kullanıcıdan elindeki tek kullanımlık parolayı sisteme girmesini isteyebilir. Böylece çok katmanlı bir güvenlik sistemi kurulabilir. Bu çalışmada tek kullanımlık parola sisteminin müşteri tarafı gerçekleştirilmiştir. Bu projede müşterilerin kullanacağı tek kullanımlık parola üreten bir cihaz gerçekleştirilmiştir. Bu konuda çalışma yapacak olanlara tek kullanımlık parola sisteminin sunucu tarafını da gerçekleştirmelerini öneriyorum. Böylece teknik olarak tam bir tek kullanımlık parolalar üreten çözüm paketi üretilmiş olur.

Tek kullanımlık parola üreten cihazların belli bir maliyeti vardır. Bilişim sistemlerinde tek kullanımlık parolalar kullanacak bir organizasyonun, örneğin müşterilerine internet üzerinden daha güvenli işlem yapabilmeleri için tek kullanımlık parola üreten cihazlar dağıtan bir banka, kurdukları sistemlerin kullanıcılarına tek kullanımlık parolalar üreten cihazlar vermeleri gerekmektedir. Bu cihazların belli bir maliyeti vardır. Ancak tek kullanımlık parolaların sağladığı yüksek güvenlik düşünüldüğünde bu maliyetlerin karşılanmasının mantıklı olduğu görülüyor. Tek kullanımlık parolalar üreten sistemler iki temel yöntemi kullanırlar. Bunlardan biri zaman senkronizayonu yöntemidir. Zaman senkronizasyonu yönteminde üretilen tek kullanımlık parolalar o anki zaman değerine bağlıdır. Bu nedenle zaman senkronizasyonu kullanan sistemlerde bir saat devresine ihtiyaç duyulur. Bu durum güç tüketimini arttırır ve bu da cihaz ömrünü azaltır. Tek kullanımlık parolalar üretmek için bir diğer yöntem de matematiksel algoritmalar kullanmaktır. Bu projede bu yöntem tercih edilmiştir. Bu projede 3-DES algoritması tek kullanımlık parolalar üretmek için kullanılan matematiksel algoritmadır. Matematiksel algoritmalar kullanan yönteme göre tek kullanımlık parolalar sadece bir önceki üretilen parolaya ve matematiksel bir fonksiyona bağlıdır. Bu nedenle kullanıcıların elindeki cihazlar parola üretileceği zamanlarda açılır ve bunun dışında kalan zamanlarda kapalı tutulursa güç tüketimi ciddi anlamda azalır ve sistem ömrü uzar.

Bu projede tasarlanan çözüm çevre faktörleri açısından incelendiğinde en önemli konulardan birinin kullanım ömrü dolan cihazların geri dönüşümü olduğu görülür. Bankaların ve devletlerin bilişim sistemlerinin milyonlarca kullanıcıları vardır. Her kullanıcıya tek kullanımlık parola üreten cihazlar verilmesi yüksek oranda malzeme tüketilmesine neden olur. Tek kullanımlık parola üreten bir cihazın kasasının geri dönüştürülebilir bir malzemeden

yapılması önemlidir. Bu projede donanım olarak MC9S12C32 mikrodenetçisini kullanan CSM12C32 geliştirme kartı kullanılmıştır. Deneyde LCD gösterge olarak HD44780 uyumlu 2x16 karakterlik arkadan aydınlatmalı bir LCD gösterge kullanılmıştır. CSM12C32 geliştirme kiti ve LCD gösterge arasındaki bağlantı 40 damarlı ara bağlantı kablosu kullanılarak yapılmıştır. Gerçekte böyle bir cihaz bir PCB baskı devre kartı üzerinde sadece MC9S12C32 mikrodenetçisi, yeterli sayıda karakter gösterebilen küçük bir LCD gösterge, bir titreşimsiz kullanıcı düğmesi ve bir güç ünitesi kullanılarak gerçekleştirilebilir. Ancak bu çalışmada hazır bulunabilen malzemeler kullanılarak bir tasarım gerçekleştirilmiştir. Bu nedenle gerçek bir cihaz bu projede gerçekleştirilen tasarıma göre çok daha az malzeme kullanır.

8 KAYNAKLAR

Rapor içinde atıfta bulunulan referanslar aşağıda bulunmaktadır.

[1] E. Adalı, *Mikroişlemciler Mikrobilgisayarlar*, Birsen Yay. 1998.

[2] Wikimedia Foundation, “One-time password”, 2013, http://en.wikipedia.org/wiki/One-time_password

[3] Wikimedia Foundation, “Security token”, 2013, http://en.wikipedia.org/wiki/Security_token

[4] Wikimedia Foundation, “Triple DES”, 2013, http://en.wikipedia.org/wiki/Triple_DES

[5] Wikimedia Foundation, “Block Cipher”, 2013, http://en.wikipedia.org/wiki/Block_cipher

[6] J. Orlin Grabbe,”The DES Algorithm Illustrated”, 2006, http://page.math.tu-berlin.de/~kant/teaching/hess/krypto-ws2006/des.htm

[7] Wikimedia Foundation, “Data Encryption Standard”, 2013, http://en.wikipedia.org/wiki/Data_Encryption_Standard