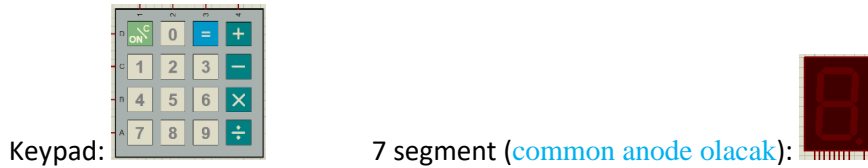


(Yeşil renkli kısımlar işinizi kolaylaştırmak için verilen bilgilerdir. Bu bilgiler de dahil, bu dokümandaki tüm istenen koşullar, (özellikle mavi kısımlar, sağlanmalıdır.)

**Soru:** Handshaking sinyalleri ile tek yönlü veri gönderme ve alma ile ilgili basit bir hesap makinesi oluşturmanız beklenmektedir. 1 giriş (keypad) ve 2 çıkış (7 segment ile 16 segment) elemanlarını kullanmanız beklenmektedir. İki ayrı 8255 ile I/O kontrolü sağlayacaksınız. Birinci 8255'e keypad ve 7 segment bağlı olacak ve *handshake* ile veri alma-gönderme yapılacak (Mod1). İkinci 8255'e 16 segment bağlanacak ve PortA & PortB birarada output olarak bu göstergeyi kontrol edecek (Mod0). Birinci 8255 için PortA 200H adresinden başlayarak ardışık olarak çift adreste, ikinci 8255 için PortA 60H adresinden başlayarak ardışık olarak çift adreste işlem yapılacaktır.

Birinci 8255'e bağlı 7 segment ve keypad için; Mod1'de PortB veri çıkışı (7 segment bağlı), PortA veri girişi (keypad bağlı) olacak şekilde tasarınızı tamamlayınız. PortC uçlarının *handshaking* için kullanılması gerektiğini hatırlayınız.



İkinci 8255'e bağlı 16 segment Mod0'da çalışacak ve PortC pinleri hiç kullanılmayacak.



Gerçeklenecek olan devre tek basamaklı 1 işlem yapan basit bir *mod* hesaplama devresidir. Tek işlem keypaddeki '+' tuşuna, mod işlemi olarak tanımlanmalıdır (Örneğin;  $3 \bmod 2 = 1 \rightarrow 3 + 2 = 1$ ). Sistem aşağıdaki gibi çalışmaktadır:

1. Keypad den girilen 1. sayı değeri alınır alınmaz *hatalı* ve *hatasız operand kontrolü*'ne tabi tutulmalıdır. Buna göre;

**Hatalı operand:** -7, -56, \*4, /99, 35, 5678, 8-, 89x, ... gibi negatif olan '-' ve sayı harici karakterle başlayan '\*', '/', '+', birden fazla basamaklı olan , tek basamaklı olsa bile devamında '+' dan farklı operator basılı olan '-' '/' '\*' kullanıcı girişleridir ve;

**1.1.** 7 segmente 0 yazdırılıp; 16 segmente ise sırasıyla 'E', 'R', 'R', 'O', 'R' harfleri aralarda kısa süreli *delay* verilerek yazdırılır (DELAY PROSEDÜRÜ). Sonda tüm segmentler (hem 7 hem 16) söndürülür, yeni baştan işlem için beklenir.

**Hatasız operand:** 3+, 5+, gibi negatif veya farklı operator başlangıcı ile girilmemiş, sadece 1 basamaklı, ve tek basamak sayıdan sonra peşi sıra '+' operator tuşuna basılmış olan kullanıcı girişleridir ve;

**1.2.** Tek basamaklı sayısal değer 7 segmente yazdırılır; ardından 16 segmente sırasıyla 'O', 'K', 'A', 'Y' harfleri aralarda kısa süreli *delay* verilerek yazdırılır (DELAY PROSEDÜRÜ). Sonda hala 7 segmentte sürekli sayısal değer gösterilmeye devam ederken, 16 segment OKAY mesajının ardından söndürülüp 2. operand için beklenir;

**1.2.1.** Hatasız operand & operator durumunda ikinci sayıya basıldığında (burada kontrol yok; sadece 1 sayıya, doğru şekilde basacağız) ise basılan sayıyı göstermeden direkt mod işleminin sonucu 7 segmentte bir süreliğine yazdırılır. Sonuca ilişkin hata kontrolüne ve 16 Segment'e bir şey yazdırmaya gerek yoktur, 16 Segment sönük kalmaya devam edecektir. (Sonuç zaten negatif çıkamaz ve 2. operanddan (tek

basamaktan) büyük olamaz.) Mod sonucu 7 segmentte bir süre gösterildikten sonra (3-5 sn.) o da sönerek yeni işlem için beklenir.



**Not1:** Flagları kullanmayı ihmal etmeyin.


Algoritma akışı için kolaylık olsun diye örnek;



*operand operator operand* = sonuç  $\rightarrow 3 \bmod 2 = 1 \rightarrow 3 + 2 = 1$


1. Veri (1. operand)  $\rightarrow 3$     2. Veri (operator)  $\rightarrow +$     3. Veri (2. operand)  $\rightarrow 2$     sonuç  $\rightarrow 1$


**Not2:** *Algoritma akışı:* Öncelikle bir tuşa basıldığını anlamamız gerekli, ilgili değeri birinci 8255 ile handshake kullanarak keypadden aldıktan sonra, hata analizi yapmalısınız:


Sayısal değer ise sorun yok . Ama ilk gelen veri operator tuşu ise (+ - \* /) o zaman hatalı operanddır .


*'0' değeri handshake ile 7 segmente bastırılır. İkinci 8255'e bağlı 16 segmentte ERROR mesajı görüntülenir. Son olarak hem 7 segment hem 16 segment sönük şekilde, yeni baştan işlem için beklenir (en başa dön).* 

Alınan ilk veri operator değil, sayı ise, yani sorun şimdilik yok ise, yukarıdaki hata kısmı olmadan 2. veri beklenir . Handshake ile keypadten alınan 2. veri yine sayı (overflow basamak) veya '+' dışında bir operator ise (- \* /) tekrar üstteki hata adımları uygulanır .

*'0' değeri handshake ile 7 segmente bastırılır. İkinci 8255'e bağlı 16 segmentte ERROR mesajı görüntülenir. Son olarak hem 7 segment hem 16 segment sönük şekilde yeni baştan işlem için beklenir.* 

Handshake ile alınan 2. veri uygun operator ise, yani '+' ise, yukarıdaki hata kısmı olmadan, 3. veriyi (yani 2. operandı) beklemek üzere olduğumuzu belirtmek için aşağıdaki adımlar koşar .

*Handshake ile alınan ilk sayısal veri (1. operand) değeri handshake ile 7 segmente bastırılır. İkinci 8255'e bağlı 16 segmentte OKAY mesajı görüntülenir. 7 segment 1. operandı gösteriyorken ve 16 segment OKAY'in ardından sönmüşken, 2. operand beklenmeye devam edilir* .

Handshake ile alınan 3. veri (2. operand) herhangi bir kontrol olmadan, 7 segmentte gösterilen 1. operand ile *mod* işlemine tabi olarak sonuç 7 segmente handshake ile gönderilerek bir süre orada gösterildikten sonra, 7 segment söner ve yeni işlem için beklenilir. Sonuca ilişkin kontrol ve bu adımda 16 segmente ilişkin bir aksiyon yoktur (zaten bir önceki adımdan sönük idi). 2. operandın hiçbir zaman ekrana basılmadığına dikkat ediniz .

Birinci 8255'te her *veri al-gönder* adımlarının handshake'e uygun olması gerekli.

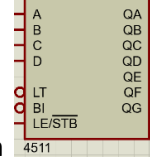
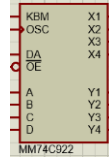
Kolaylık olması için örnek bir DELAY PROC  $\rightarrow$

```
DELAY PROC NEAR
PUSH CX
MOV CX, 05FFFh
COUNT:
LOOP COUNT
POP CX
RET
DELAY ENDP
```

'E' DELAY 'R' DELAY 'R' DELAY 'O' DELAY 'R' DELAY

**Not3:** Handshake kısmı derste anlatılan kısım gibi. Kolay, ara devreleri (NOT kapısı olan yapıdan bahsediyorum) oradan bakarak yapabilir; kodlayabilirsiniz.

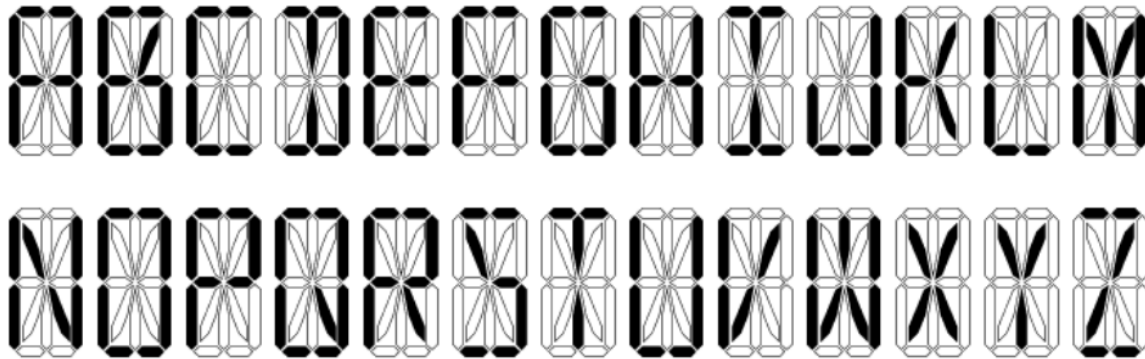
**Not4:** Key paddeki On/C tuşu ve '=' işareti önemsizdir, kullanmanıza gerek yok.









**Not5:** Keypad için yardımcı devresini, 7 segment için yardımcı devresini kullanmayı unutmayın; hem handshake için gerekli, hem de işlerinizi kolaylaştıracak. 16 segment Mod0'da çalışan ayrı bir 8255'e direkt bağlı olacağı için ek eleman gerektirmiyor (yardımcı donanım KESİNLİKLE KULLANILMAYACAK; pinler direkt Port A & B ile sürülecek). İkinci 8255'in PortA ve PortB tüm pinlerini kullanmanız gerekecek (PortA ve PortB output; PortC hiç kullanılmayacak). Bu segmente veri gönderirken, göndereceğiniz veriyi, örneğin "T" harfini, 16 segmentin yanan ve yanmayan bacaklarına göre belirleyip, HEX değeri elde ettikten sonra PortA ve PortB'ye uygun yarı değerleri göndereceksiniz; burada yaptığınız bağlantı da önemli olacak. Segment pinleri PortA'nın / PortB'nin hangi pinlerine bağlı? (MSB – LSB sıralaması vb.)



16 segmente ilişkin harf tablosunu şöyle gösterebiliriz:



SERCAN =      

Segmentin *Common Cathode* olmasını göz önüne almayı unutmayınız. Bacak numaraları için internette kısa bir araştırma yapınız.

**Not6:** Devre 1 defa aritmetik işlemi yapınca 2. defa başka bir aritmetik işlemi yapmak için yeniden başlatılmak zorunda kalırsa da sorun yok; yeter ki 1 defa kontrollü olarak aldığı birinci operanda ikinciyi MOD işlemi ile uygulasin; mesajları & sonucu doğru gösterecek. Sonraki işlemler için simülasyonu durdurup, yeniden açsak da sıkıntı yok.

