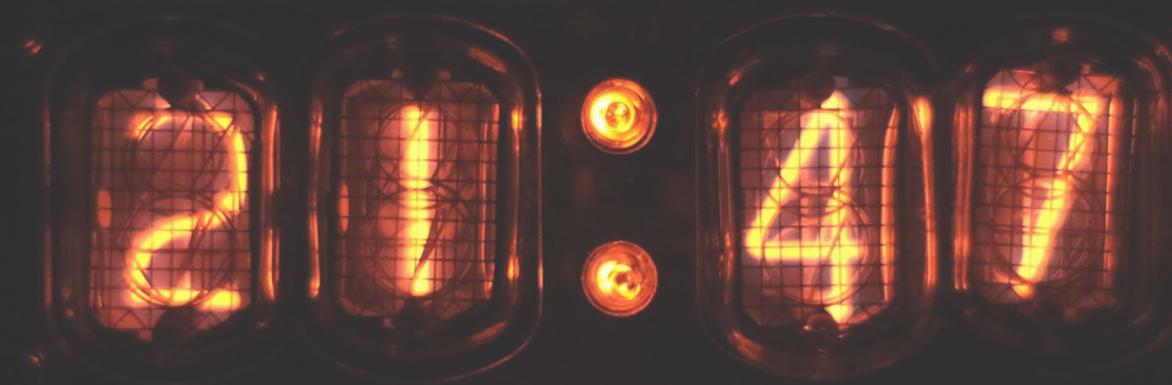


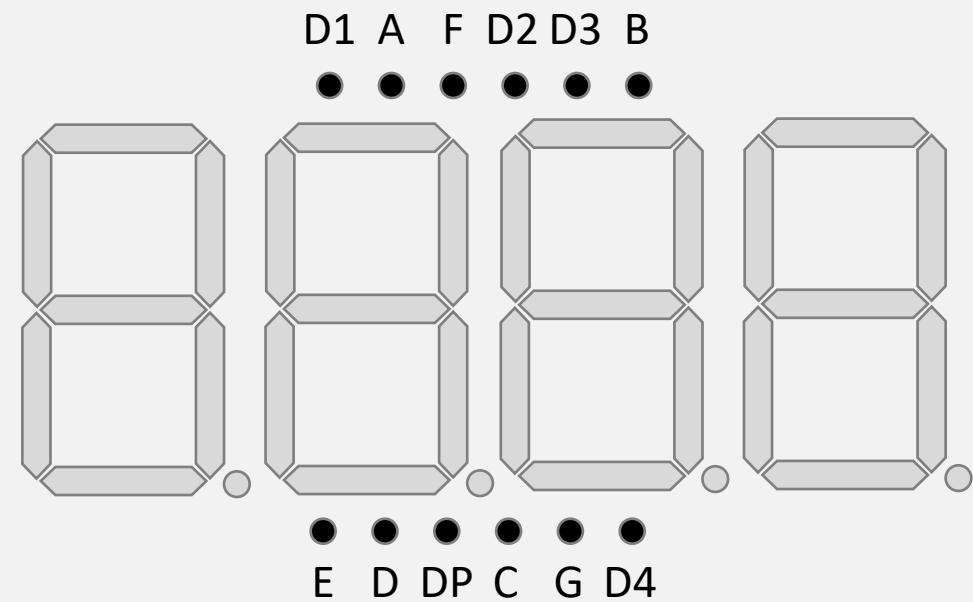
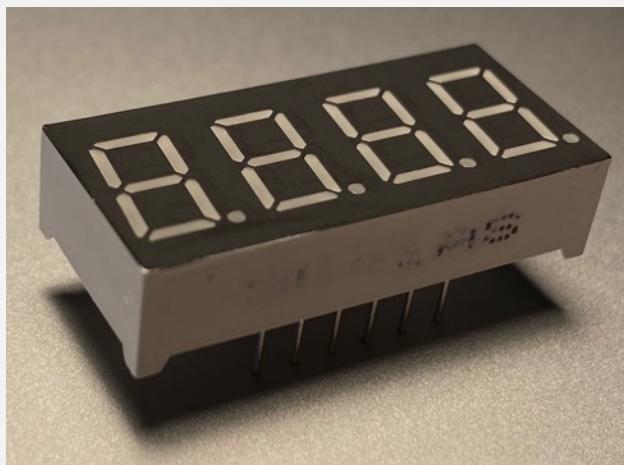
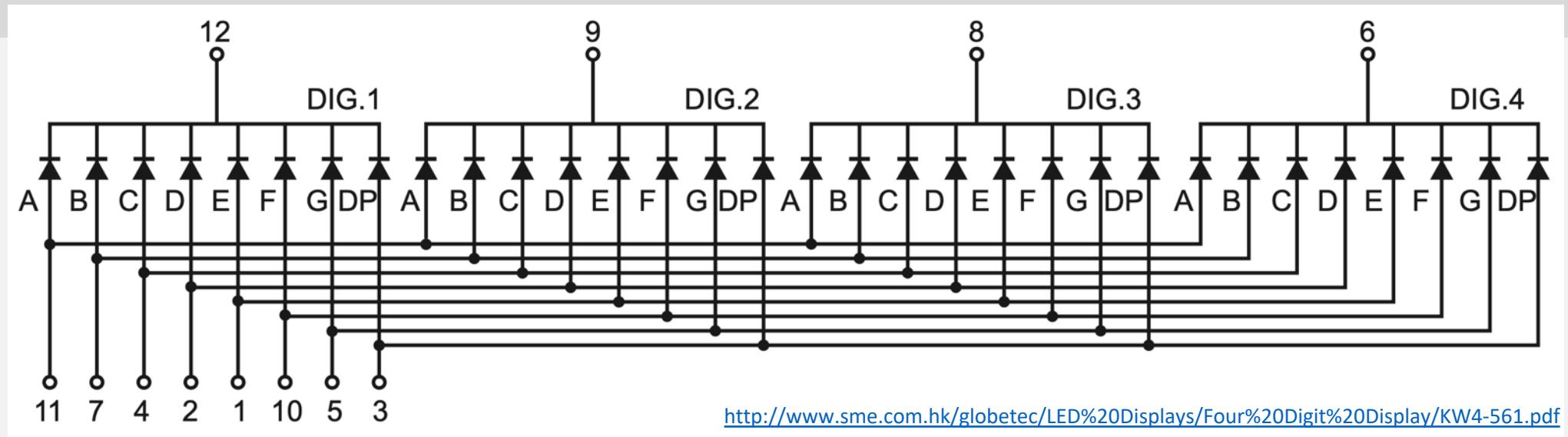
4 digits



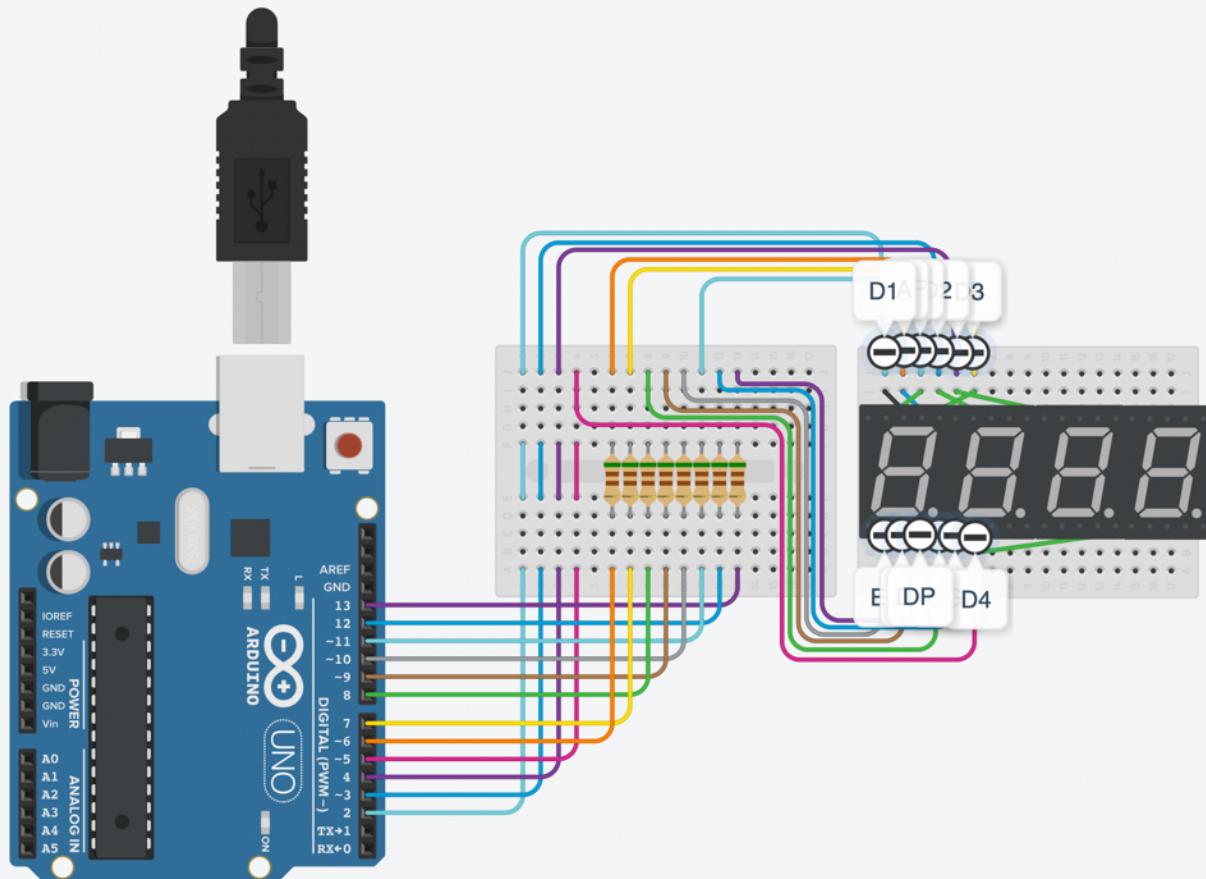
# Objectifs

- Présentation du 4 digits
- Montage et programme de test
- Multiplexage
- Gérer le temps

# Présentation – 3461AS



# Programme de test



```
int segmentCourant = 0;  int digitCourant = 0;

void setup() {
    for (size_t pin = 2; pin <= 13; ++pin) {
        pinMode(pin, OUTPUT);
    }
    reset();
}

void loop() {
    digitalWrite(segmentCourant + 6, HIGH);
    digitalWrite(digitCourant + 2, LOW);

    delay(500);

    digitalWrite(segmentCourant + 6, LOW);
    digitalWrite(digitCourant + 2, HIGH);

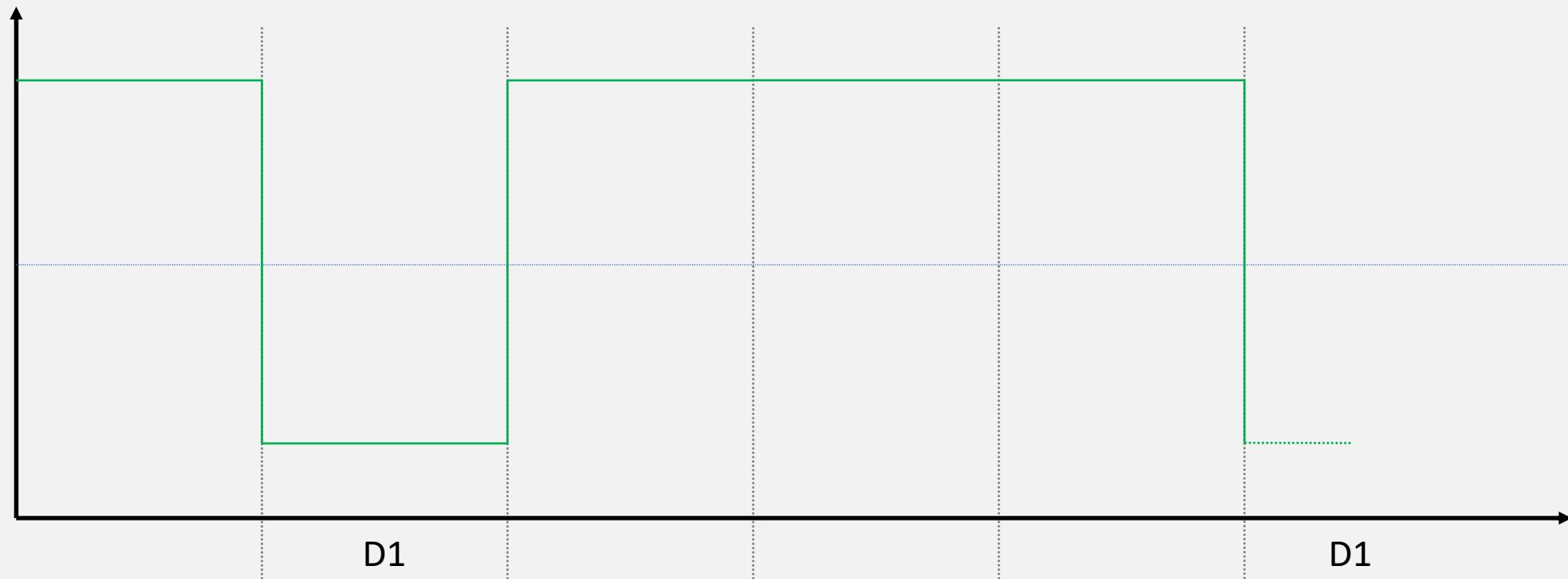
    if (++segmentCourant >= 8) {
        segmentCourant = 0;
        digitCourant = (digitCourant + 1) % 4;
    }
}

void reset() {
    for (size_t pin = 2; pin <= 5; ++pin) { digitalWrite(pin, HIGH); }

    for (size_t pin = 6; pin <= 13; ++pin) { digitalWrite(pin, LOW); }
}
```

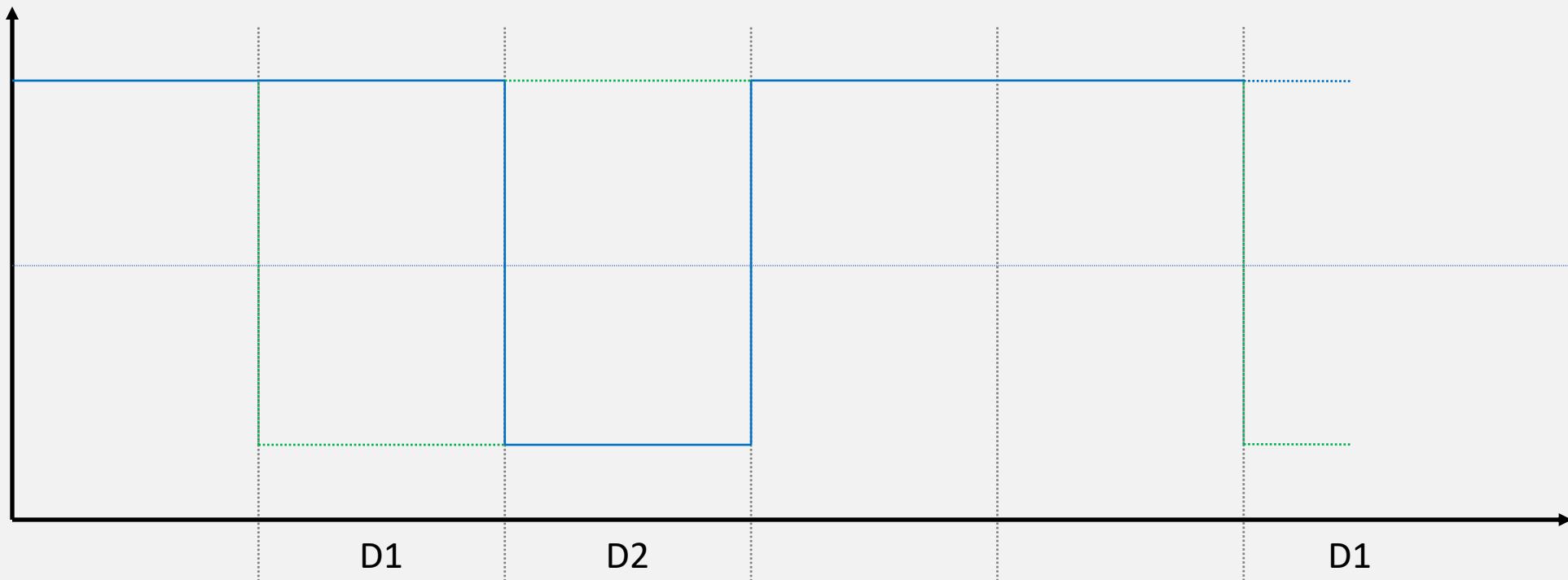
# Comment afficher les 4 digits - Multiplexage

- Idée :
    - Allumer tous les digits, un digit à la fois pendant une durée très courte
    - Répéter l'opération plusieurs fois par seconde : la persistance rétinienne va donner l'illusion que le digit est allumé



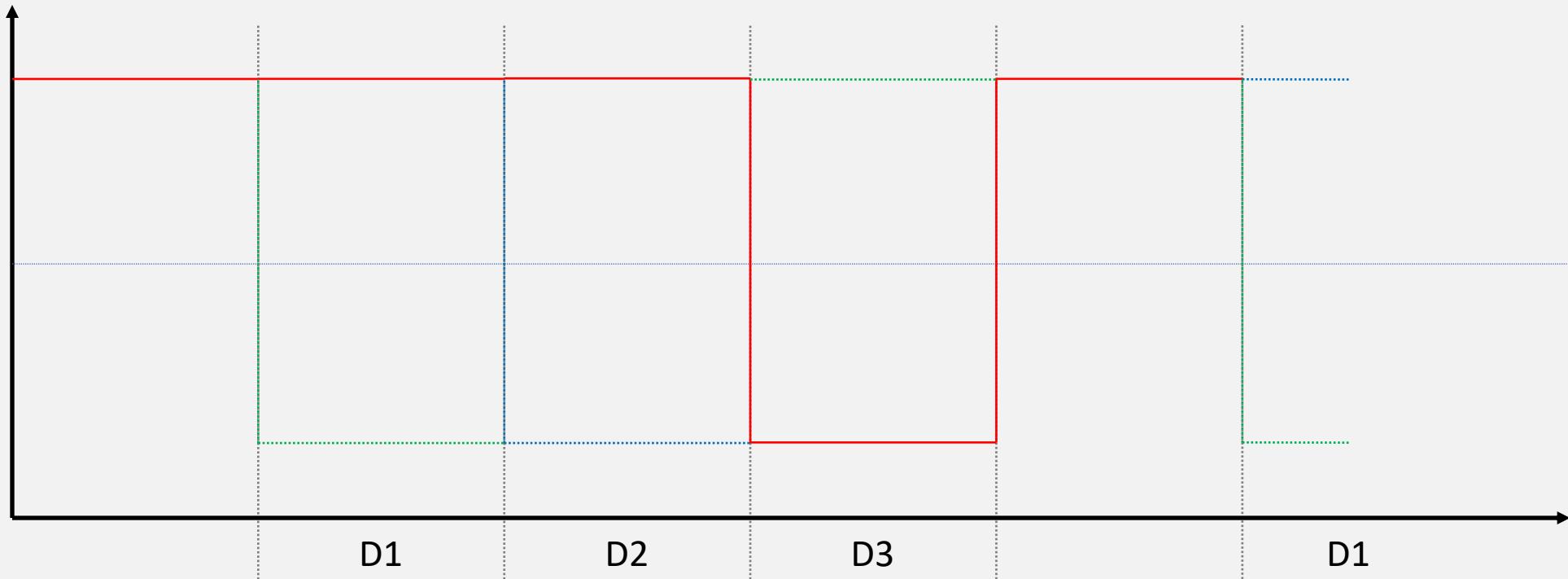
# Comment afficher les 4 digits - Multiplexage

- Idée :
  - Allumer tous les digits, un digit à la fois pendant une durée très courte :
  - Répéter l'opération plusieurs fois par seconde : la persistance rétinienne va donner l'illusion que le digit est allumé



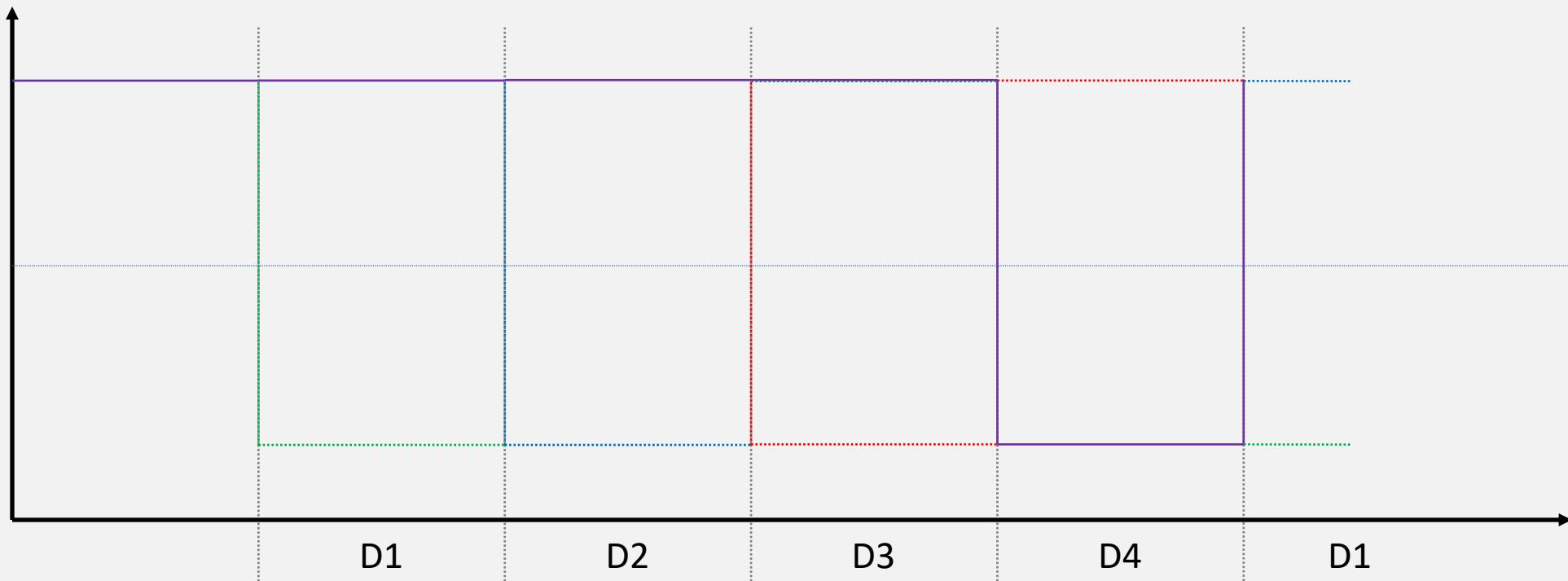
# Comment afficher les 4 digits - Multiplexage

- Idée :
  - Allumer tous les digits, un digit à la fois pendant une durée très courte :
  - Répéter l'opération plusieurs fois par seconde : la persistance rétinienne va donner l'illusion que le digit est allumé



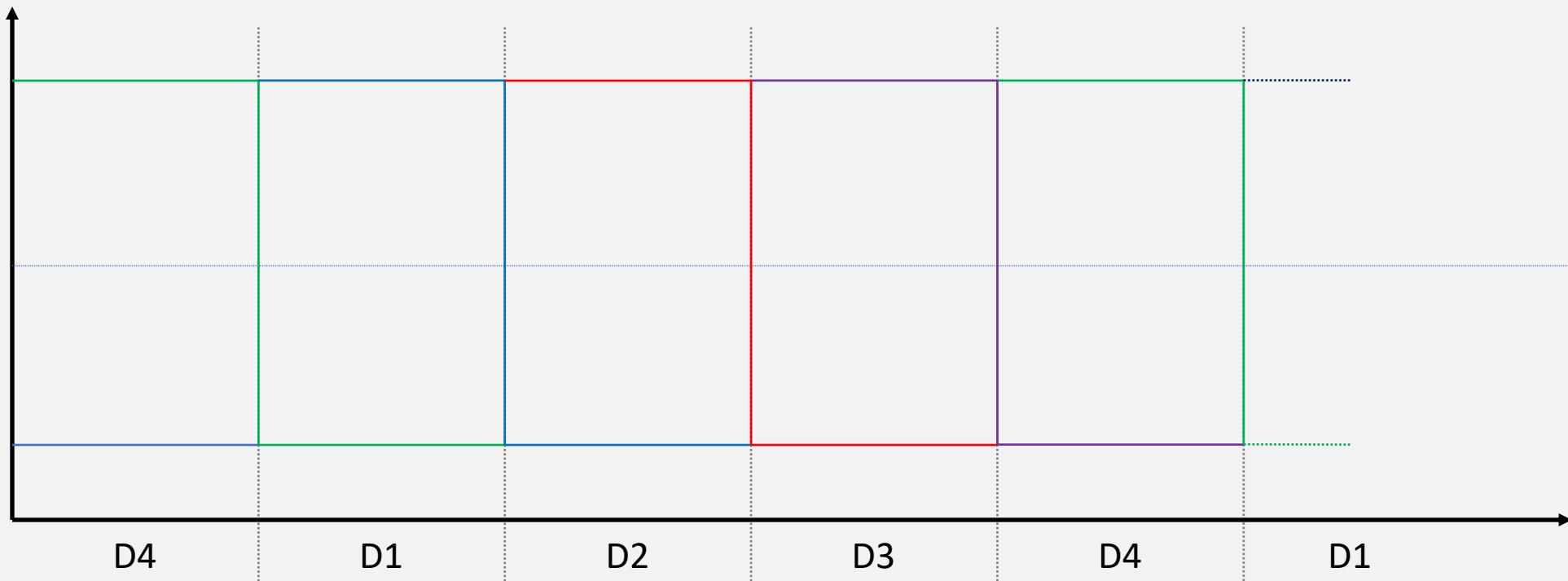
# Comment afficher les 4 digits - Multiplexage

- Idée :
  - Allumer tous les digits, un digit à la fois pendant une durée très courte :
  - Répéter l'opération plusieurs fois par seconde : la persistance rétinienne va donner l'illusion que le digit est allumé



# Comment afficher les 4 digits - Multiplexage

- Idée :
  - Allumer tous les digits, un digit à la fois pendant une durée très courte :
  - Répéter l'opération plusieurs fois par seconde : la persistance rétinienne va donner l'illusion que le digit est allumé



# Comment afficher les 4 digits - Multiplexage

- Idée :
  - Allumer tous les digits, un digit à la fois pendant une durée très courte :
  - Répéter l'opération plusieurs fois par seconde : la persistance rétinienne va donner l'illusion que le digit est allumé



# Comment afficher les 4 digits - Multiplexage

- Plusieurs façons de faire pour contrôler la durée d'affichage de chaque digit :
  1. Boucle principale avec des validations de durée : utilisation de « millis » ou « micros »
  2. Utilisation d'une interruption basée sur le temps

# Affichage 4 digits basé sur la boucle principale (loop)

```
class Affichage4DigitsDirect {  
public:  
    Affichage4DigitsDirect(const int& p_pinSegmentA, [...]p_pinSegmentDP,  
                           const int& p_pinD1, [...]p_pinD4,  
                           const unsigned long &p_dureeAffichageDigit);  
  
    void Afficher(const int &p_valeurDigit1, [...]p_valeurDigit4) const;  
  
    void Executer() const;  
  
private:  
    void EnvoyerValeur(const byte p_valeur) const;  
  
private:  
    int m_segmentOff; int m_segmentOn;  
    int m_digitOff; int m_digitOn;  
  
    int m_pinD[4];  
    int m_pinSegments[8];  
  
    volatile mutable byte m_cache[4];  
    volatile mutable int m_digitCourant;  
    volatile mutable unsigned long m_microsDernierChangement;  
  
    const int m_dureeAffichageDigit;  
};
```



```
Affichage4DigitsDirect adr(6, 7, 8, 9,  
                           10, 11, 12, 13,  
                           2, 3, 4, 5);  
  
void setup() {  
    adr.Afficher(1, 2, 3, 4);  
}  
  
void loop() {  
    adr.Executer();  
}
```

# Affichage 4 digits basé sur la boucle principale (loop)

```
void Affichage4DigitsDirect::Executer() const {
    unsigned long horloge = micros();

    if (this->m_microsDernierChangement + this->m_dureeAffichageDigit < horloge) {
        this->m_microsDernierChangement = horloge;
        digitalWrite(this->m_pinD[this->m_digitCourant], this->m_digitOff);

        this->m_digitCourant = (this->m_digitCourant + 1) % 4;

        this->EnvoyerValeur(this->m_cache[this->m_digitCourant]);

        digitalWrite(this->m_pinD[this->m_digitCourant], this->m_digitOn);
    }
}
```

# Affichage 4 digits basé sur la boucle principale (loop)

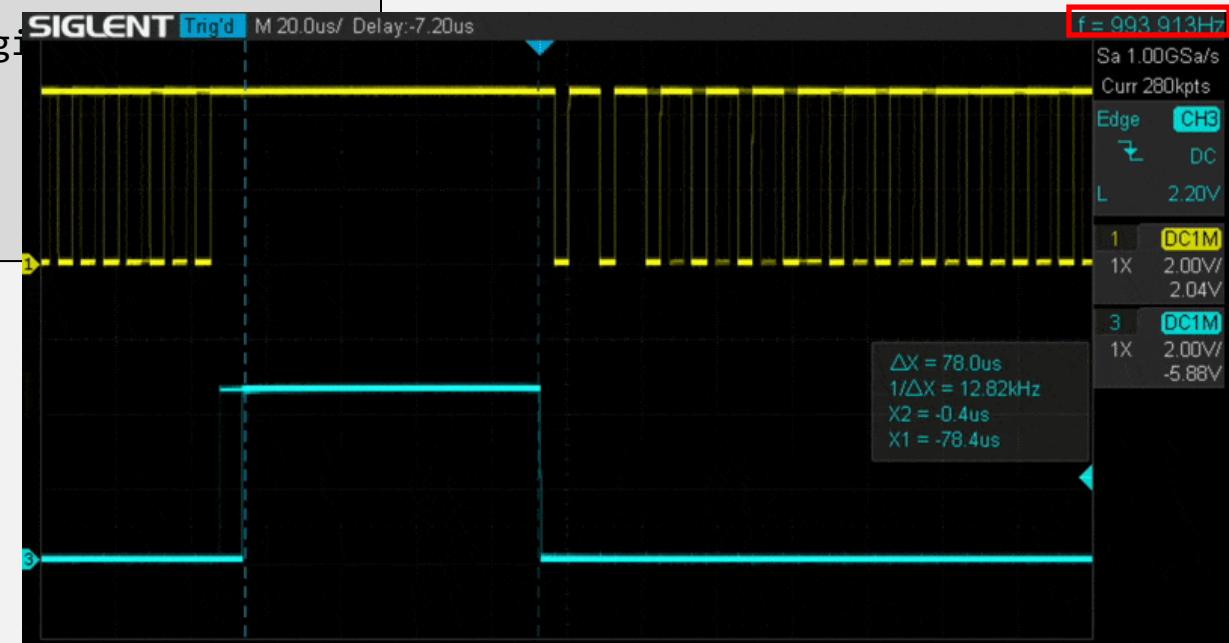
```
void Affichage4DigitsDirect::Executer() const {
    unsigned long horloge = micros();

    digitalWrite(A0, HIGH);
    if (this->m_microsDernierChangement + this->m_dureeAffichageDigit < horloge) {
        digitalWrite(A1, HIGH);
        this->m_microsDernierChangement = horloge;
        digitalWrite(this->m_pinD[this->m_digitCourant], this->m_digitOff);

        this->m_digitCourant = (this->m_digitCourant + 1) % 4;

        this->EnvoyerValeur(this->m_cache[this->m_digitCourant]);

        digitalWrite(this->m_pinD[this->m_digitCourant], this->m_digitOn);
        digitalWrite(A1, LOW);
    }
    digitalWrite(A0, LOW);
}
```



# Affichage 4 digits basé sur les interruptions

```
cli();
TCCR1A = 0;
TCCR1B = 0;
TCNT1 = 0;
// doit être < 65536
// Pour 1000Hz (16*10^6) / (64 * 1000) - 1
OCR1A = 249;
// turn on CTC mode
TCCR1B |= (1 << WGM12);
// Déf CS10 et CS11 bits pour 64 prescaler
TCCR1B |= (1 << CS11) | (1 << CS10);
// Activer la comparaison pour l'interruption
TIMSK1 |= (1 << OCIE1A);
sei();
```

```
ISR(TIMER1_COMPA_vect)
{
    adr.Executer();
}
```

# Affichage 4 digits basé sur les interruptions

```
void Affichage4DigitsDirect::Executer() const {
    // unsigned long horloge = micros();

    digitalWrite(A0, HIGH);
    // if (this->m_microsDernierChangement + this->m_dureeAffichageDigit < horloge) {
        digitalWrite(A1, HIGH);
        // this->m_microsDernierChangement = horloge;
        digitalWrite(this->m_pinD[this->m_digitCourant], this->m_digitOff);

        this->m_digitCourant = (this->m_digitCourant + 1) % 4;

        this->EnvoyerValeur(this->m_cache[this->m_digitCourant]);

        digitalWrite(this->m_pinD[this->m_digitCourant], this->m_digitOn);
        digitalWrite(A1, LOW);
    // }
    digitalWrite(A0, LOW);
}
```



# Références

- <https://www.tinkercad.com/things/7Ha0tayTvkk> : Programme de test et montage de base
- <https://www.tinkercad.com/things/gQ3CXT26zba> : Programme pour afficher 1234
- <https://www.youtube.com/watch?v=ZXcmeEMK730> : Affichage 4 digits – Cyrob partie 1
- <https://www.youtube.com/watch?v=ZXcmeEMK730> : Suite (avancée) - Cyrob partie 2