自動貯金箱の製作

藤井　優奈　大門　新菜

１．研究概要

現在の社会では、多くのセンサが利用されている。しかし、私たちはどのセンサがどのような用途に使われているのかについては、十分に理解できていないのが現状である。

そこで私たちは、センサを利用している身近なもののなかで、硬貨の判別を行うために利用されているセンサに目をつけた。貯金箱は主に硬貨の保管に利用されており、通常の貯金箱では確認することが困難な合計金額の把握を行う貯金箱を製作することにした。自動貯金箱の製作を通してセンサの理解を深めることができるとともに、ソフトウェアの新たな知識を身につけるための研究を行った。

２．研究の具体的内容

（１）硬貨の大きさの詳細

表１は、硬貨の直径の大きさを示したものである。

表１　硬貨の直径

１円玉	20.0mm
50 円玉	21.0mm
５円玉	22.0mm
100 円玉	22.6mm
10 円玉	23.5mm
500 円玉	26.5mm

（２）動作説明

硬貨を投入する
↓
表２を元に硬貨の直径で仕分けられる
（１円、50 円、５円、100 円、10 円、500 円）
↓
硬貨がフォトインタラプタを通過した回数を集計し、合計金額を計算する
↓
７セグメント LED に合計金額を表示する
（写真１）

※引き出しが引かれると合計金額がリセットされる仕組みになっている

図１　動作の流れ

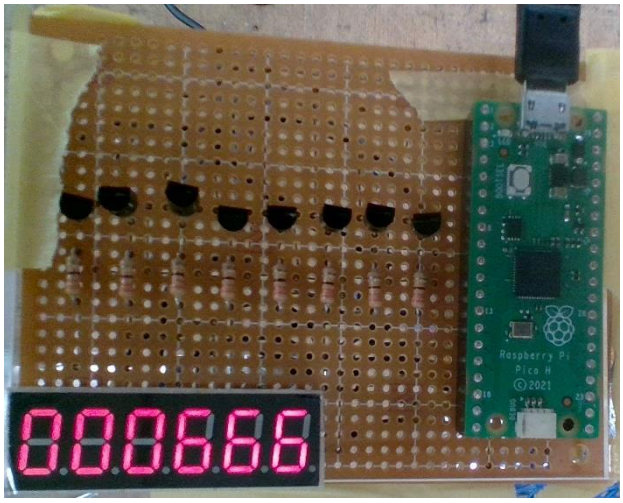

写真１　合計金額の表示

（３）使用材料

表２　部品一覧

部品名	個数
フォトインタラプタ [CNZ1023(ON1023)]	7
Raspberry Pi Pico H	1
６桁７セグメントLED表示機 [OSL60362-IR]	1
トランジスタ [2SC1815]	8
抵抗（330Ω）	15
抵抗（10kΩ）	7
ユニバーサル基盤 (95mm × 72mm)	2
ピンソケット	1
アクリル板（全て２mm ）	1
L字金具	8
丁番	4
取手	2

硬貨の選別には磁気センサやリミットスイッチを使用する案もあったが、磁気センサは汚れがあると反応しづらく、リミットスイッチは長期間使用するとはずみが生じることからフォトインタラプタが最適だと判断し、フォトインタラプタを使用することにした。

（４）箱の製作

まず、貯金箱の大まかな形状や寸法を決定するために、仕分け部分をアクリルで作り、その他の部分を段ボールで構成した仮の貯金箱を製作した（写真２、３）。箱の大きさは表３の通りである。

当初、仕分け部分は傾きのみを利用して硬貨の選別を行っていたが、指定の場所を通り過ぎてしまうことがあり、角度をつけて落ちやすくなるように工夫した。

仮の段階では、仕分け部分に筒がなく、引き出しは３分割していた。しかし、硬貨が落ちた時に指定の場所に収まらないという問題が発生したため、最終的に大きな引き出しを製作し、硬貨が引き出し内に収まるよう工夫した。仕分け部分については、段ボールを使用すると硬貨の滑りが悪くなるため、仮の段階からアクリル板を使用した。しかし、段ボールにはアクリル用接着剤を使用できなかったため、仕分け部分は天井部分に穴を開け、針金で吊り下げる形を取った。

表３　寸法

部品名	寸法 縦 × 横 × 高さ(mm)
箱全体	250 × 350 × 300
引き出し	120 × 350 × 70
仕分け部分	6 × 360 × 60
筒	90 × 30 × 15

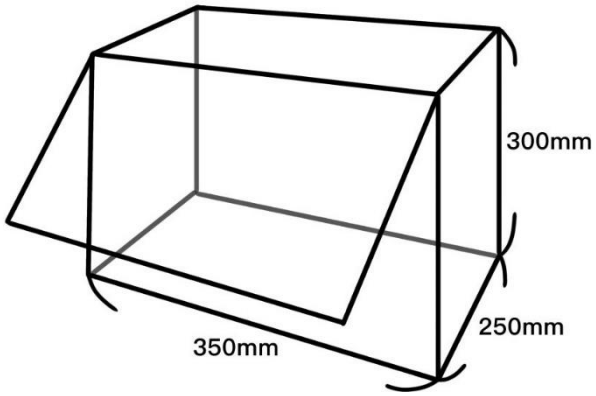

図２　箱全体の設計図

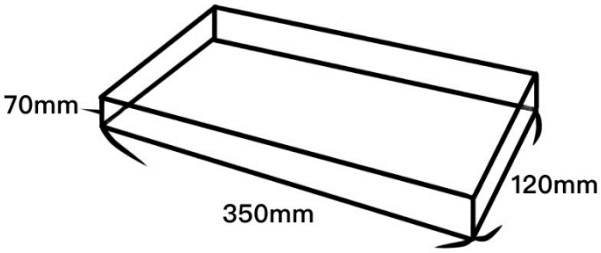

図３　引き出しの設計図

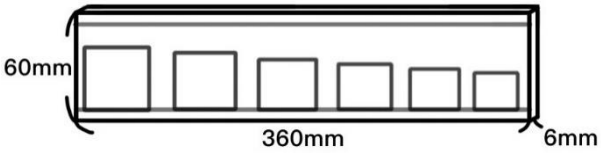

図４　仕分け部分の設計図

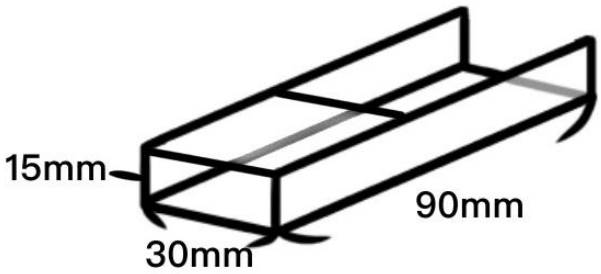

図５　筒の設計図

図５の蓋は硬貨の空振りを防ぐために試行を重ねた結果、最も空振りが少なかった 30mm にした。

写真２　箱１（仮）

写真３　箱２（仮）

仮の貯金箱を基に以下の手順で箱の製作を行った（写真４）。

１．Fusion 360 を利用して、切断するアクリル板の図面を表４、５の寸法通りに作成する。
２．レーザー加工機を利用してアクリル板を切断する。
３．キリでネジを通す穴を開ける。
４．L 字金具や丁番をネジで固定する。
　箱の前面と後面のみ丁番で固定し、
　取手を取り付けて開閉できるようにしている。
５．仕分け部分と引き出し部分のアクリル板をアクリル用接着剤で固定する。

表４　箱の寸法

図面名	縦 × 横（mm）
底面・上面	250 × 350
左側面・右側面	296 × 246
前面・後面	350 × 246

表５　引き出しの寸法

図面名	縦 × 横（mm）
底面	120 × 330
左側面・右側面	70 × 116
前面・後面	70 × 330

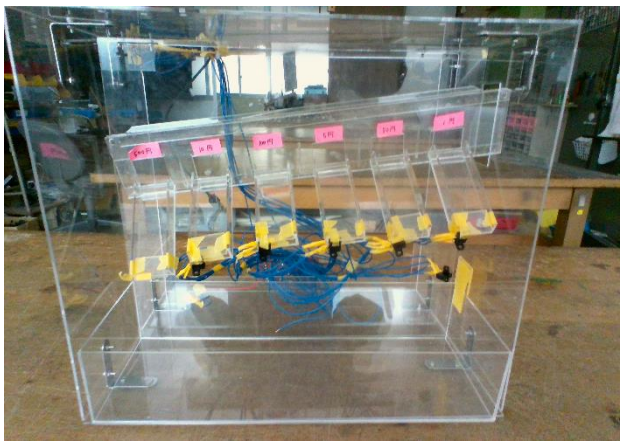

写真４　箱（完成版）

表３を参考にアクリル板を切断したが、アクリル板の厚さを含んでいなかったことで上下でのサイズの違いが生じ、寸法を修正しながら箱の作成を何度か行った。

仕分け部分は、各硬貨の直径（表１）× 30mm の間隔で穴を配置し、投入された硬貨が小さい順にそれぞれの穴を通るように設計している。しかし、硬貨の落下位置が異なる問題や、連続して硬貨を投入すると滑りが悪くなる問題が生じた。これらの問題に対して、仕分け部分の斜辺や角度を調整することで、改善を図った。硬貨が通過した先は筒状になっており、フォトインタラプタを確実に通過する設計となっている（写真５）。通過した硬貨は、そのまま引き出し内へ落下する。当初、箱の組み立てはアクリル用接着剤のみでの予定だったが、安定性がなかったためＬ字金具を利用して組み立てた。

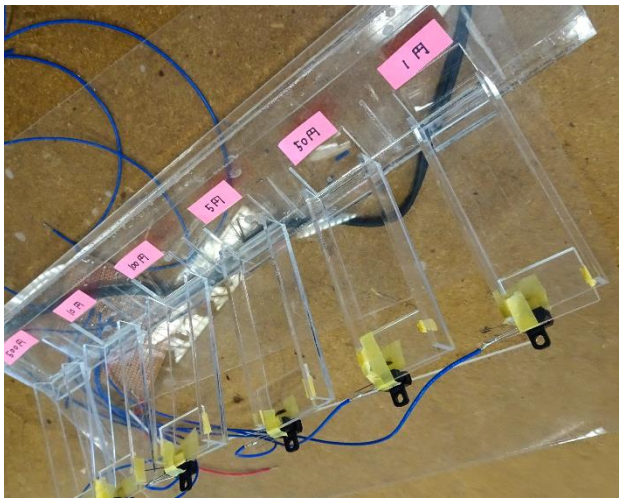

写真５　仕分け部分

（５）回路設計・配線

ブレッドボードを使用し、各部品の回路を確認しながら、以下の回路図（図７，８，９）を基に回路を作成した。作成した回路から実体配線図を作成し、95mm × 72mm のユニバーサル基板を２枚使用して各部品を取り付けた。

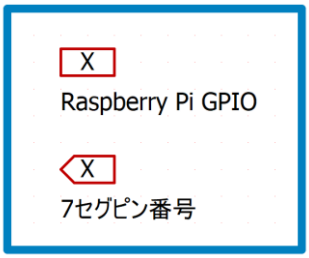

図６　ピン番号

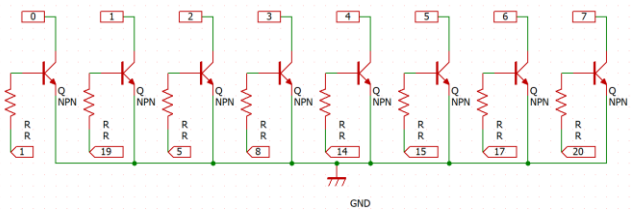

図７　回路図１

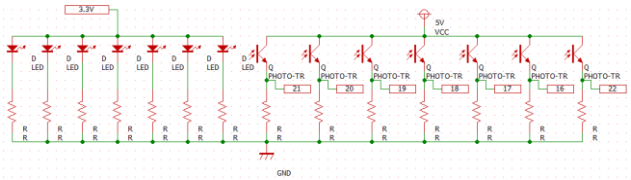

図８　回路図２

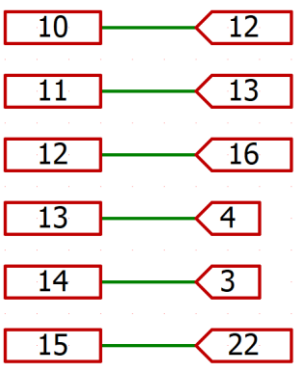

図９　回路図３

金額表示の基板には、トランジスタ、６桁７セグメントLED、Raspberry Pi Pico H を取り付けた（写真６，７）。硬貨検知の基板には、フォトインタラプタ回路で使用する抵抗を取り付けた（写真８，９）。基板内で配線が可能な部分についてはすずメッキ線を用い、それ以外の配線にはビニール絶縁電線を使用して空中配線を行った。

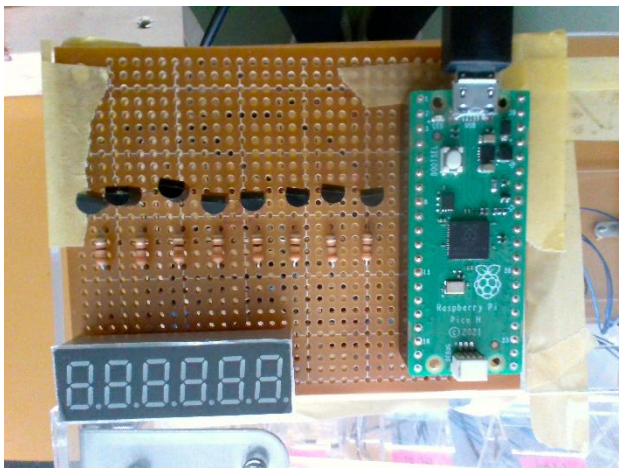

写真６　金額表示回路

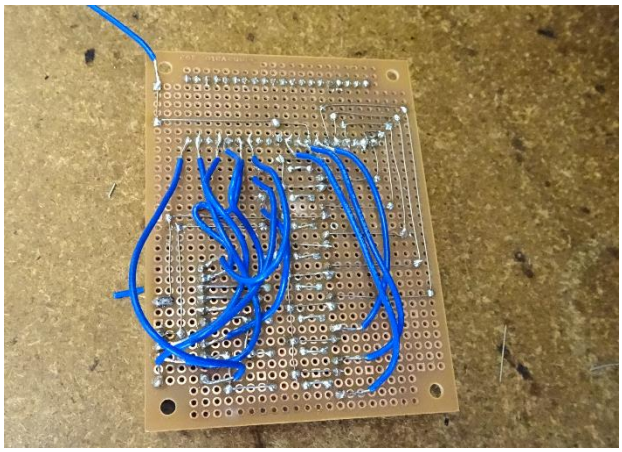

写真７　金額表示回路（配線）

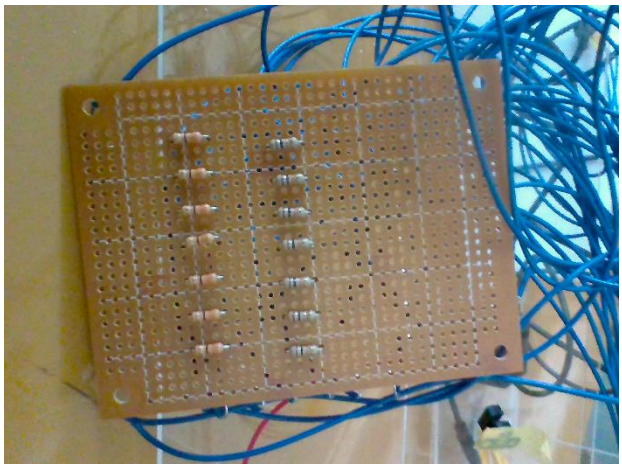

写真８　硬貨検知回路

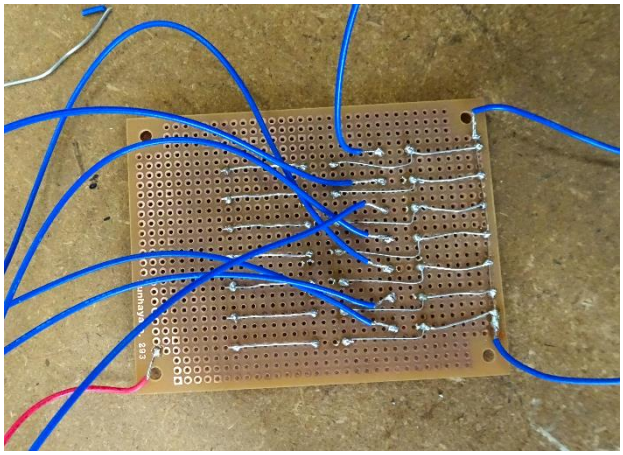

写真９　硬貨検知回路（配線）

電源は２種類あり、１つはコンセントからmicroUSB ケーブルを使用してRaspberry Pi Pico H に接続する 3.3V のもの、もう１つはスイッチング AC アダプタを利用してコンセントから供給する５V のものである。５V の電源については、DC ジャックを使用して配線できるようにした（写真 10)。

写真 10　電源部分

（６）プログラム

プログラムは Python を使用し、Raspberry Pi Pico H の書き込みに対応した Thonny を用いて作成した。if 文を使って判定した後、それぞれのフォトインタラプタに対応した計算を行い、最終的に value 関数に合計を代入して表示する実装にした。

```
import machine
import time
```

図 10　関数の import

machine 関数は Raspberry Pi Pico H との通信に必要な関数であり、time 関数は時間を扱うための関数である。この２つの関数を import することで、Raspberry Pi Pico H との連携を行うことができる。

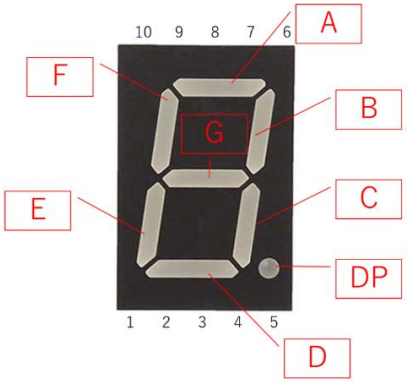

写真 11　７セグメント LED のピン割り付け

```
numbers = [
    #E,  #C,  #D,  #DP,  #F,  #A,  #G,  #B
    [1,   1,   1,   0,    1,   1,   0,   1], #0
    [0,   1,   0,   0,    0,   0,   0,   1], #1
    [1,   0,   1,   0,    0,   1,   1,   1], #2
    [0,   1,   1,   0,    0,   1,   1,   1], #3
    [0,   1,   0,   0,    1,   0,   1,   1], #4
    [0,   1,   1,   0,    1,   1,   1,   0], #5
    [1,   1,   1,   0,    1,   1,   1,   0], #6
    [0,   1,   0,   0,    1,   1,   0,   1], #7
    [1,   1,   1,   0,    1,   1,   1,   1], #8
    [0,   1,   1,   0,    1,   1,   1,   1]  #9
]
```

図 11　数字表示の配列の定義

Numbers 配列は、写真 11 をもとに光らせる部分を１、光らせない部分を０として、それぞれの数字を表示する配列となっている。

```
def clear():
machine.Pin(0,machine.Pin.OUT).value(0)
machine.Pin(1,machine.Pin.OUT).value(0)
machine.Pin(2,machine.Pin.OUT).value(0)
machine.Pin(3,machine.Pin.OUT).value(0)
machine.Pin(4,machine.Pin.OUT).value(0)
machine.Pin(5,machine.Pin.OUT).value(0)
machine.Pin(6,machine.Pin.OUT).value(0)
machine.Pin(7,machine.Pin.OUT).value(0)
machine.Pin(10,machine.Pin.OUT).value(0)
machine.Pin(11,machine.Pin.OUT).value(0)
machine.Pin(12,machine.Pin.OUT).value(0)
machine.Pin(13,machine.Pin.OUT).value(0)
machine.Pin(14,machine.Pin.OUT).value(0)
machine.Pin(15,machine.Pin.OUT).value(0)

a1 = 0
a2 = 0
a3 = 0
a4 = 0
a5 = 0
a6 = 0
b = 0
c = 0
d = 0
e = 0
f = 0
g = 0
value = 0
```

図 12　初期値設定

図 12 は、使用している Pin の初期設定と、計算で使用する関数の初期値を設定したものである。

```
while True:
    if machine.Pin(22, machine.Pin.IN)
                            .value() == 0:
        a1 = 0
        a2 = 0
        a3 = 0
        a4 = 0
        a5 = 0
        a6 = 0
    elif machine.Pin(22, machine.Pin.IN)
                            .value() == 1:
        if machine.Pin(16, machine.Pin
                        .IN).value() == 0:
            if b == 0:
                a1 += 1
                b = a1
        elif machine.Pin(16, machine.Pin
                        .IN).value() == 1:
            b = 0
        if machine.Pin(17, machine.Pin
                        .IN).value() == 0:
            if c == 0:
                a2 += 50
                c = a2
        elif machine.Pin(17, machine.Pin
                        .IN).value() == 1:
            c = 0
        if machine.Pin(18, machine.Pin
                        .IN).value() == 0:
            if d == 0:
                a3 += 5
                d = a3
        elif machine.Pin(18, machine.Pin
                        .IN).value() == 1:
            d = 0
```

図 13　計算処理（前半部分）

```
        if machine.Pin(19, machine.Pin
                        .IN).value() == 0:
            if e == 0:
                a4 += 100
                e = a4
        elif machine.Pin(19, machine.Pin
                        .IN).value() == 1:
            e = 0
        if machine.Pin(20, machine.Pin
                        .IN).value() == 0:
            if f == 0:
                a5 += 10
                f = a5
        elif machine.Pin(20, machine.Pin
                        .IN).value() == 1:
            f = 0
        if machine.Pin(21, machine.Pin
                        .IN).value() == 0:
            if g == 0:
                a6 += 500
                g = a6
        elif machine.Pin(21, machine.Pin
                        .IN).value() == 1:
            g = 0
```

図 14　計算処理（後半部分）

図 13 の while 文は、それぞれのフォトインタラプタが反応した時の計算を行うプログラムである。Pin 番号 22 番は引き出しを引いたときのリセット用であり、すべての値をリセットする役割を果たす処理になっている。

```
  value = a1 + a2 + a3 + a4 + a5 + a6
    for n in range(6):
        for i in range(8):
            machine.Pin(0+i,
machine.Pin.OUT).value(numbers[int(value
% 10)][i])
```

```
        machine.Pin(10+n,
            machine.Pin.OUT).value(1)
        time.sleep(0.001)
        machine.Pin(10+n,
            machine.Pin.OUT).value(0)
        value = value / 10
```

図 15　合計金額の計算・表示

図 15 は合計金額を計算する部分のプログラムである。time.sleep は、７セグメントディスプレイの点滅間隔を指定するために使用されている。

３．岡工祭の展示について

岡工祭では、3.3V は通常通り Raspberry Pi Pico から電源を供給したが、５V 電源には専用のスイッチアダプタがなかったため、電源装置を利用した。自動貯金箱としては動作したものの、仕分け部分の製作が間に合わず、フォトインタラプタをテープで仮止めしていたため、硬貨が確実にフォトインタラプタを通過する動作ができず、加算される場合とされない場合があった。

４．個人のまとめ

藤井　優奈

今回、春休みから課題研究を行ってきたが、３年間の知識では足りないことに気づくことができた。そのため知識の習得時間をスケジュールに含んでおらず、３年生前半から大きく遅れをとってしまった。しかしながら、後半からは効率的に作業を行えるようになり、無事岡工祭に間に合ったということに大きな達成感を感じたとともに、挑戦することの大切さを学ぶことができたと感じている。

大門　新菜

自動貯金箱を制作するにあたり、まず年間計画を立てた。行事や進路に関わる予定を考慮した計画を立てたつもりだったが、実際に作業を進める中で、計画通りに進めることの難しさを痛感した。就職活動や部活動の影響で作業に参加できず、藤井さんに任せきりになることも多く、申し訳ない気持ちでいっぱいだった。岡工祭の直前には、時間ぎりぎりまで残って作業を進め、何とか形にすることができて安心した。しかし、完成した自動貯金箱にはまだ多くの改善点があるため、今後さらに改良を重ねていきたいと考えている。

５．研究のまとめ

センサを利用して小銭の判別を行うため、Raspberry Pi Pico やプログラミング言語 Python を用いてプログラムを作成した。３年間で学んだことを応用しながら、新しい物事にも積極的に挑戦することができた。スケジュール通りに進めることが難しく、放課後に残って作業を進めることが多かったが、最終的には一通りの動作を実現し、Python などの知識も身に付けることができた。貯金箱の機能には合計金額をスマートフォンに表示するといった改善すべき点が残されているため、今後さらに改良を加えていきたいと考えている。

参考文献

- Thonny の使い方
  https://sanuki-tech.net/and-more/2022/install-thonny-python-ide/

- 複数桁の７セグメント LED を動かす方法
  https://qiita.com/umet787X/items/517819406e962320fe4d