

**Digitální Hodiny**

Ročníková práce z předmětu PSS

Matyáš Prokop

C3B

# Obsah

[**Obsah**](#_tehz6jsmi4lv) **2**

[**Zadání Práce**](#_8qggo2e5atox) **3**

[**Úvod**](#_ndko6eineatm) **4**

[**Ekonomická rozvaha**](#_6jj8k6k4hs58) **5**

[Konkurence](#_8vjqz86k23sd) 5

[Výhody mého řešení](#_z8uj3ggq9t7m) 5

[Propagace](#_du8qr92wgs8e) 5

[Návratnost investic](#_dfkjeqfhkhi7) 5

[**Rozbor**](#_hlil4zljdkgb) **6**

[Architektura systému](#_3ijkb0yhfptd) 6

[Funkční toky:](#_zer8sofcyv9d) 6

[Hardware](#_w7qc0oegq2ff) 6

[Software](#_8k8n5cear298) 7

[Arduino kód](#_uz5gdc6qbxtr) 7

[Klíčové části:](#_yf590fyq6plw) 7

[Silné stránky:](#_n24ab7ixwk2p) 7

[Python skript](#_770r9xoy8cbl) 7

[Funkce:](#_2sx4l6ze8nq5) 7

[Klíčové části:](#_arx24ajvkxop) 8

[Výhody:](#_afkx8tsv4s7) 8

[Komunikace mezi Pythonem a Arduinem](#_9ot49yfb406n) 8

[Výhody:](#_z6kki7g33lr6) 8

[**Realizace**](#_plzo26k9zj16) **9**

[Zapojení pinů](#_t8qwwvhta9xw) 9

[Programování Arduina](#_afby2411m90o) 10

[Python skript a API komunikace](#_vx6mh12rav36) 10

[Integrace systému](#_aq8usbrnox8r) 11

[Problémy a řešení](#_9ky51qbesjpw) 11

[**Vyhodnocení**](#_ukmlzfkk7sjn) **11**

[**Jak spustit projekt?**](#_ogq33pydv31) **14**

[Instalace Pythonu](#_4nx97xjpg2hw) 14

[Nainstalujte potřebné knihovny](#_fyz09piv9yj) 14

[Nastavení Google API](#_u5qu7cu29lb) 14

[Spusťte Python kód](#_n27hxiha6afr) 15

[Vytvořte události v Google Kalendáři](#_nsddho3ira0j) 15

[Instalace Arduino IDE a knihoven](#_b0kyt8wvlugu) 15

[Zapojení a spuštění](#_5k0w8pq1olp0) 16

[**Testovací Scénáře**](#_vj9ymnvxnsjf) **17**

[**Seznam obrázků**](#_7ladv1dzdaaq) **20**

# 

# 

# Zadání Práce

Cílem této práce bylo vytvořit digitální hodiny, které střídavě zobrazují aktuální čas a datum. Mezi přechody těchto údajů byla navržena animace, která zajišťuje plynulý a esteticky příjemný přechod mezi zobrazením času a schůzek. V pozdější fázi byl projekt rozšířen o propojení s Google Kalendářem prostřednictvím Google Calendar API. Toto propojení je realizováno pomocí Python skriptu, který umožňuje načítání událostí z uživatelova kalendáře a jejich následné zobrazení na displeji hodin.

# 

# Úvod

Moderní technologie nám umožňují propojit každodenní digitální nástroje s fyzickým světem. V rámci této práce byl vyvinut systém digitálních hodin, který kromě zobrazování běžného času a data umožňuje také zobrazovat nejbližší události z kalendáře uživatele. Využitím platformy Arduino spolu s RTC modulem (DS3231), TM1637 LED displejem a propojením na Google Kalendář pomocí Python skriptu a oficiálního API vzniklo funkční zařízení, které může pomoci s každodenním plánováním.

Zařízení je navrženo s důrazem na jednoduchost, rozšiřitelnost a nízkou cenu. Přidaná hodnota spočívá v propojení s online kalendářem, čímž se zařízení stává interaktivním a reaguje na aktuální plán uživatele. Zvolený hardware je běžně dostupný a projekt je navržen tak, aby ho mohl sestavit a používat i začátečník.

# Ekonomická rozvaha

## **Konkurence**

Na trhu existují produkty jako chytré hodiny (např. od Apple nebo Xiaomi), případně chytré domácí asistenty s displejem (Google Nest Hub, Amazon Echo Show). Tyto produkty však:

* Jsou výrazně dražší (v řádech tisíců Kč)
* Vyžadují složitější konfiguraci
* Neumožňují jednoduché vlastní úpravy

## **Výhody mého řešení**

* Nízké pořizovací náklady (do 500 Kč)
* Otevřený kód a snadná úprava
* Lokální provoz bez nutnosti stálého připojení k internetu (čas z RTC)
* Možnost zobrazení jen relevantních událostí, bez rušivých notifikací

## **Propagace**

Projekt lze propagovat formou:

* Příspěvků na GitHubu, Hackaday.io
* Sociálních sítí
* Zapojení do školních či soutěžních prezentací

## **Návratnost investic**

Díky otevřenosti projektu a nízkým nákladům lze předpokládat rychlé přijetí mezi hobby komunitou. Projekt může sloužit jako výukový nebo rozšiřitelný základ, jehož hodnota přesahuje pouhé zařízení – zejména při učení API a práce s HW.

# Rozbor

## **Architektura systému**

Projekt kombinuje embedded systém s online cloudovou službou, konkrétně Google Kalendářem, a propojuje je pomocí Python skriptu běžícího na počítači a Arduino zařízení. Celý systém funguje ve dvou hlavních rovinách:

1. Mikrokontrolér Arduino + displej + RTC modul
2. Python skript pro API komunikaci s Google Kalendářem

## Funkční toky:

* **Arduino**: Lokálně zobrazuje čas, datum a události na LED displeji.
* **Python skrip**t: Stahuje nejbližší události z Google Kalendáře a posílá je přes sériový port do Arduina.
* **Zobrazení událostí**: Arduino třídí, ukládá a postupně zobrazuje jednotlivé události, přičemž využívá i jednoduchou animaci pro přechody.

## **Hardware**

**Arduino Uno -** Hlavní mikrokontrolér, zajišťuje zpracování a zobrazení dat

**RTC DS3231** - Hodinový modul s vysokou přesností a funkcí záložního napájení

**TM1637 LED Display** - 4-místný 7 segmentový displej pro zobrazení času a událostí

**Seriové spojení -** Komunikace s počítčaem, přenos dat z Python Skriptu

## **Software**

### Arduino kód

#### **Klíčové části:**

* **setup()**: Inicializace RTC modulu, displeje a paměti pro události.
* **loop()**: Hlavní smyčka volá getIncomingTime() a aktualizuje časovače.
* **getIncomingTime()**: Čeká na vstup po sériové lince ve formátu eventMinutes,currentTimeInMinutes.
* **event\_arr\_t**: Vlastní implementace dynamického pole událostí.
* **showTime() / showDate()**: Zobrazení aktuálního času a data.
* **mainDisplayCycle()**: FSM (stavový automat) pro střídání času, datumu a událostí.

#### 

#### 

#### **Silné stránky:**

* Použití timer knihovny pro elegantní správu časových událostí bez delay().
* Animace mezi přechody hodnot zvyšuje interaktivitu.
* Dobrá modularizace funkcí a rozdělení podle úloh.

### Python skript

#### **Funkce:**

* Autentizace přes OAuth 2.0 s uložením tokenu.
* Připojení k **Google Calendar API**.
* Získání nejbližších 10 událostí (čas i název).
* Výpočet událostí v minutách a jejich odeslání přes COM port Arduinu.

#### **Klíčové části:**

* sendToArduino(start, summary):  
  + Parsuje čas začátku události.
  + Vypočítá minuty od půlnoci.
  + Posílá string ve formátu eventMinutes,currentTimeInMinutes.
* main():  
  + Inicializace připojení a získání dat.
  + Zobrazení ladicích informací.

#### **Výhody:**

* Využití oficiálního API.
* Ukládání tokenu pro budoucí přístup.
* Nízká latence přenosu a jednoduchost formátu.

## Komunikace mezi Pythonem a Arduinem

* Použit je sériový port (COM3) s rychlostí 9600 baudů.
* Data jsou zasílána v jednoduchém formátu jako text, oddělený čárkou.
* Arduino očekává celý řetězec a rozparsuje čas události.

#### **Výhody:**

* Přehledný a jednoduchý formát.
* Možnost připojení i jiných zařízení, pokud umí generovat podobný výstup.

# Realizace

Samotná realizace projektu byla rozdělena do dvou hlavních částí: návrh a sestavení hardwarové části s Arduinem a implementace softwarového propojení s Google Kalendářem pomocí Pythonu. Všechny části byly navrženy tak, aby je bylo možné snadno reprodukovat s běžně dostupnými komponenty.

## Zapojení pinů

| **Komponenta** | **Arduino Pin** |
| --- | --- |
| TM1637 CLK | D2 |
| TM1637 DIO  TM1637 GND  TM1637 VCC | D3  GND  3.3V |
| RTC SDA  RTC SCL  RTC VCC  RTC GND | A4  A5  5V  GND |

Napájení je realizováno přes USB port počítače.

**Obrázek 1 - zapojení arduinaZapojení Hodin
**

## Programování Arduina

Arduino firmware je napsán v jazyce C++ a zajišťuje tři hlavní funkce:

* **Zobrazení času a data:** Informace z RTC modulu jsou zobrazeny na displeji pomocí knihovny TM1637Display. Čas je zobrazen s dvojtečkou, datum bez ní.
* **Zobrazení událostí:** Události přicházejí přes sériový port ve formátu eventMinutes,currentTimeInMinutes a jsou uloženy do dynamického pole. Následně jsou zobrazeny v cyklu po jedné.
* **Animace přechodu:** Mezi přechody (např. mezi časem a událostmi) probíhá jednoduchá animace posouvání segmentů LED displeje, která zvyšuje atraktivitu výstupu.

Hlavní logika zobrazení je řízena pomocí časovače (Timer knihovna), který volá stavový automat pro cyklické přepínání mezi časem, datem a seznamem událostí.

## Python skript a API komunikace

Druhou část systému tvoří **Python skript**, který běží na počítači. Ten se připojuje k Google Kalendáři pomocí oficiálního API a OAuth 2.0 autentizace.

Hlavní kroky skriptu:

1. **Autentizace:** Při prvním spuštění proběhne přihlášení uživatele a vytvoření token.json pro další použití.
2. **Získání událostí:** Pomocí Google Calendar API se stáhne seznam nejbližších 10 událostí.
3. **Převod a přenos:** Každá událost je převedena na počet minut od půlnoci a odeslána na Arduino ve formátu eventMinutes,currentTimeInMinutes.

Tento jednoduchý textový protokol bez potvrzovacího mechanismu je plně dostačující pro základní jednosměrnou komunikaci.

## Integrace systému

Po spuštění Python skriptu se data o událostech odešlou do Arduina, které je uloží a v pravidelném cyklu zobrazuje. Systém funguje i bez internetového připojení – pokud nejsou dostupná nová data, zobrazuje čas a datum ze svého RTC modulu. V případě výpadku napájení si RTC modul udrží aktuální čas.

## Problémy a řešení

Během vývoje bylo nutné řešit několik drobných problémů:

* **Synchronizace událostí** – přijatá data bylo potřeba filtrovat a třídit podle času.
* **Paměťová náročnost** – použití dynamického pole vyžadovalo ruční správu paměti, což bylo řešeno funkcemi pro realokaci a třídění.
* **Přesnost RTC** – modul DS3231 se ukázal jako velmi stabilní, ale bylo nutné zohlednit možnost ručního nastavení při ztrátě napájení.

# Vyhodnocení

Projekt naplnil původní cíle – vytvořit funkční digitální hodiny s možností zobrazení nejbližších událostí z Google Kalendáře. Během realizace se ukázalo několik klíčových přínosů i výzev:

Silné stránky:

* **Nízké náklady:** Cena všech komponent zůstala pod 500 Kč, což výrazně odlišuje projekt od komerčně dostupných řešení.
* **Modularita a otevřenost:** Díky otevřenému kódu lze zařízení snadno upravit, rozšířit (např. o Wi-Fi nebo větší displej) nebo použít jako výukovou pomůcku.
* **Funkčnost bez internetu:** RTC modul zajišťuje přesný čas i bez aktivního připojení.
* **Jednoduché ovládání a přehlednost:** Displej zobrazuje jen to, co je důležité – bez rušivých prvků nebo notifikací.

Slabé stránky a výzvy:

* **Závislost na počítači pro aktualizaci událostí:** V aktuální verzi je nutné mít spuštěný Python skript na počítači, který se stará o komunikaci s Google API.
* **Bezpečnost a šifrování:** Přenos dat mezi Python skriptem a Arduinem není nijak šifrován ani autentizován.
* **Omezený počet znaků na displeji:** TM1637 umožňuje zobrazit pouze čtyři číslice, takže zobrazení názvů událostí nebo delších informací není možné.

Možnosti dalšího rozvoje:

* Implementace Wi-Fi (např. pomocí ESP32) pro nezávislý přístup k internetu bez nutnosti běžícího PC.
* Přechod na větší displej pro zobrazení názvů událostí.
* Vylepšené uživatelské rozhraní, např. s dotykovými tlačítky pro manuální přepínání mezi režimy.
* Zpětná kontrola přijetí dat mezi Python skriptem a Arduinem.

# 

# 

# 

# 

# 

Zkušenosti a návrhy pro vylepšení:

Z praktického hlediska jsem během práce dospěl k závěru, že při opakované realizaci nebo rozšíření projektu bych zvolil jinou platformu – konkrétně Raspberry Pi místo Arduina.

Raspberry Pi má oproti Arduinu několik zásadních výhod:

* **Vestavěné připojení k internetu (Wi-Fi nebo Ethernet):** To by odstranilo nutnost používat samostatný počítač pro komunikaci s Google Kalendářem.
* **Schopnost spouštět Python skript přímo:** Raspberry Pi má vlastní operační systém a umožňuje snadno spustit Python skript i další služby automaticky po startu zařízení.
* **Více výpočetního výkonu a paměti:** Umožnil by složitější zpracování dat, například zobrazení názvů událostí, více událostí naráz, nebo použití dotykového displeje.
* **Jednodušší bezpečnostní integrace:** Práci s OAuth, certifikáty a šifrováním by bylo možné řešit přímo v Pythonu bez zbytečné serializace a přenosu dat.

# Jak spustit projekt?

Tento projekt propojuje Google Kalendář s Arduinem a převádí události na „čas v minutách“. Níže je kompletní návod k nasazení.

## Instalace Pythonu

Nejprve si stáhněte a nainstalujte **Python verze 3.12** z oficiální stránky:  
 <https://www.python.org/downloads/release/python-3120/>

Při instalaci **zaškrtněte volbu "Add Python to PATH"**, jinak nebude fungovat v příkazové řádce.

## Nainstalujte potřebné knihovny

Otevřete terminál / příkazový řádek a nainstalujte následující knihovny:

bash

ZkopírovatUpravit

pip install google-auth google-auth-oauthlib google-api-python-client pyserial

## Nastavení Google API

Zprovozněte Google Calendar API podle oficiální dokumentace:  
 👉 Google Calendar API quickstart - Python

1. Vytvořte projekt v Google Cloud Console
2. Aktivujte **Google Calendar API**
3. Vytvořte **OAuth 2.0 Client ID**
4. Stáhněte si soubor credentials.json

Přidejte credentials.json do projektu

Ve složce s projektem najdete adresář credentials. Nahraďte jeho obsah souborem credentials.json, který jste stáhli z Google API.

**Nezapomeňte**:

* Přidat svůj Google účet do testovacích uživatelů v Google Cloud Console (jinak ověření nebude fungovat)

## Spusťte Python kód

Spusťte skript pomocí:

bash

ZkopírovatUpravit

python nazev\_souboru.py

Po prvním spuštění:

* se otevře prohlížeč s výzvou k přihlášení a ověření přístupu,
* po úspěšném ověření se vytvoří soubor token.json, který uchovává přístupový token.

## Vytvořte události v Google Kalendáři

Zadejte do svého kalendáře události s konkrétními časy. Po spuštění skriptu budou tyto události převedeny na minuty a odeslány do Arduina.

## Instalace Arduino IDE a knihoven

1. Stáhněte si **Arduino IDE** z oficiální stránky:  
    https://www.arduino.cc/en/software
2. Otevřete adresář s Arduino kódem (soubor .ino).
3. Přetáhněte potřebné knihovny do složky libraries, která se nachází ve složce Arduino, tyto knihovny najdete přiložené v projektu.

# 

## Zapojení a spuštění

* Zapojte hardware podle schématu, které bylo uvedeno v sekci *Realizace* (nezapomeňte vložit baterii do RTC modulu).
* Nahrajte kód do Arduina pomocí Arduino IDE. (nezapomeňte vybrat arduino uno)
* Ujistěte se, že máte události v Google Kalendáři a že Arduino je připojené (např. na portu COM3).
* Spusťte opět Python skript – data se převedou a odešlou do zařízení.

# Testovací Scénáře

Testovací scénář 1 – Zobrazení aktuálního času a data z RTC modulu

**Cíl:** Ověřit správnou inicializaci a fungování RTC modulu DS3231 a zobrazování času a datumu na displeji.

* **Postup:** Po zapnutí zařízení se ověřilo, že čas a datum jsou správně načteny z RTC modulu a zobrazeny na displeji v cyklu.
* **Výsledek:** ✅ Čas i datum byly zobrazeny správně a přesně. RTC si držel hodnotu i po odpojení napájení díky záložní baterii.  
   ✅ Přechody mezi časem a datem probíhaly s animací bez zadrhávání.

# 

Testovací scénář 2 – Příjem a zobrazení událostí z Google Kalendáře

**Cíl:** Otestovat komunikaci Python skriptu s Google Calendar API a následný přenos událostí do Arduina.

* **Postup:** Do Google Kalendáře byly zadány tři testovací události v různých časech. Skript byl spuštěn, provedlo se ověření účtu a data byla odeslána přes sériový port.
* **Výsledek:** ✅ Události byly správně načteny, převedeny na minuty od půlnoci a úspěšně odeslány do Arduina.  
   ✅ Na displeji se jednotlivé události střídaly podle pořadí a času.

# 

### 

### 

### 

Testovací scénář 3 – Ztráta připojení k internetu / nefunkční Python skript

**Cíl:** Ověřit chování systému v případě, že není dostupné nové API spojení.

* **Postup:** Python skript nebyl spuštěn, zařízení běželo pouze s RTC modulem.
* **Výsledek:** ✅ Hodiny i nadále zobrazovaly přesný čas a datum.  
   ✅ Nebyly zobrazeny žádné události, což odpovídá očekávání.

# 

Testovací scénář 4 – Spuštění a nasazení systému

**Cíl:** Ověřit celý proces nasazení – od instalace software po funkční zobrazení na zařízení.

* **Postup:**
  1. Nainstalován Python a potřebné knihovny.
  2. Vytvořen Google API projekt a získán credentials.json.
  3. Připojeno Arduino, nahrán firmware.
  4. Spuštěn Python skript a zadány události v kalendáři.
* **Výsledek:** ✅ Celý proces proběhl bez chyb.  
   ✅ Skript úspěšně autentizoval účet, získal události a odeslal je do Arduina.  
   ✅ Na displeji se střídaly aktuální čas, datum a jednotlivé události.  
   ✅ Projekt lze považovat za funkční a připravený pro koncové použití.

# 

## 

Testovací scénář 5 – Zobrazení více událostí v cyklu

**Cíl:** Ověřit, zda zařízení správně zobrazí více událostí z Google Kalendáře, které přijdou po sériovém portu, a zda je mezi nimi správné přepínání.

* **Postup:**

1. V Google Kalendáři jsem vytvořil 5 událostí s různými časy začátku (např. 9:00, 11:30, 13:45, 15:00 a 17:15).
2. Spustil jsem Python skript, který získal události a poslal je ve formátu eventMinutes,currentTimeInMinutes do Arduina.
3. Na displeji TM1637 jsem sledoval postupné přepínání zobrazení mezi časem, datem a jednotlivými událostmi.

* **Výsledek:** ✅ Všechny události byly úspěšně načteny a odeslány na Arduino.  
   ✅ Displej střídal zobrazení jednotlivých událostí podle očekávání.

# 

# Seznam obrázků

[1] - Obrázek zapojení arduina…. 11