به نام خدا

راهنمای ماژول Ev1527\_decoder.bas

# **معرفی**

فایل ev1527\_decoder.bas یک ماژول است که با استفاده از تایمر 1 در مد کپچر سیگنال چیپ های 1527 یا موارد مشابه را با دقت مناسبی دیکد میکند

برای جلوگیری از تداخل اسمی حین استفاده از ماژول تمامی متغیر ها، ثابت ها و برچسب های استفاده شده داخل ماژول پیشوند \_ev1527d دارند

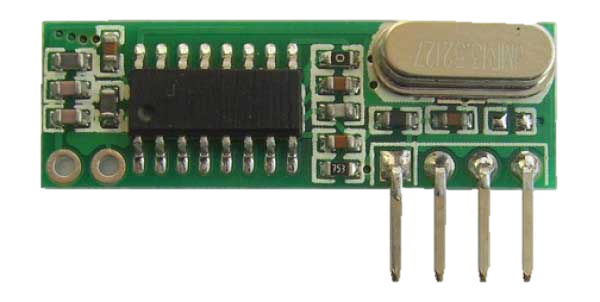
برای خوانایی بیشتر در این مقاله ، اسم ها بدون پیشوند هستند

در این فایل تنظیماتی در نظر گرفته شده که بر حسب نیاز قابل تغییر هستند. در ادامه راجع به آن ها بیشتر توضیح داده شده.

تنظیمات Prescale تایمر ها به صورت اتوماتیک بر اساس کریستال محاسبه و انتخاب میشود

این ماژول رو به راحتی میتوانید در هر پروژه ای با نوشتن چند خط اضافه کنید

از لحاظ سخت افزاری فقط کافیه خروجی ماژول مدولاتور رو به پایه ICP1 میکرو بدید و تا جای امکان ماژول به میکرو نزدیک باشه ترجیحا Track رو با پلیگان GND شیلد کنید و از مدولاتور های سوپرهتروداین استفاده کنید



# 

# **توضیحات کلی**

این ماژول متشکل از چهار مد بوده که بر حسب نیاز قابل استفاده هستند

**مد توقف (stop) :**

در این مد کلیه تایمر ها و اینتراپت ها غیر فعال و به طور کلی ماژول غیر فعال است.

**مد شناسایی (detect) :**

این مد ساده ترین حالت است و هر بار کدی شناسایی میشود سابروتین detected اجرا و کد دریافتی از طریق آرایه 3 بایتی Code قابل خواندن خواهد بود.

**مد امن (safe) :**

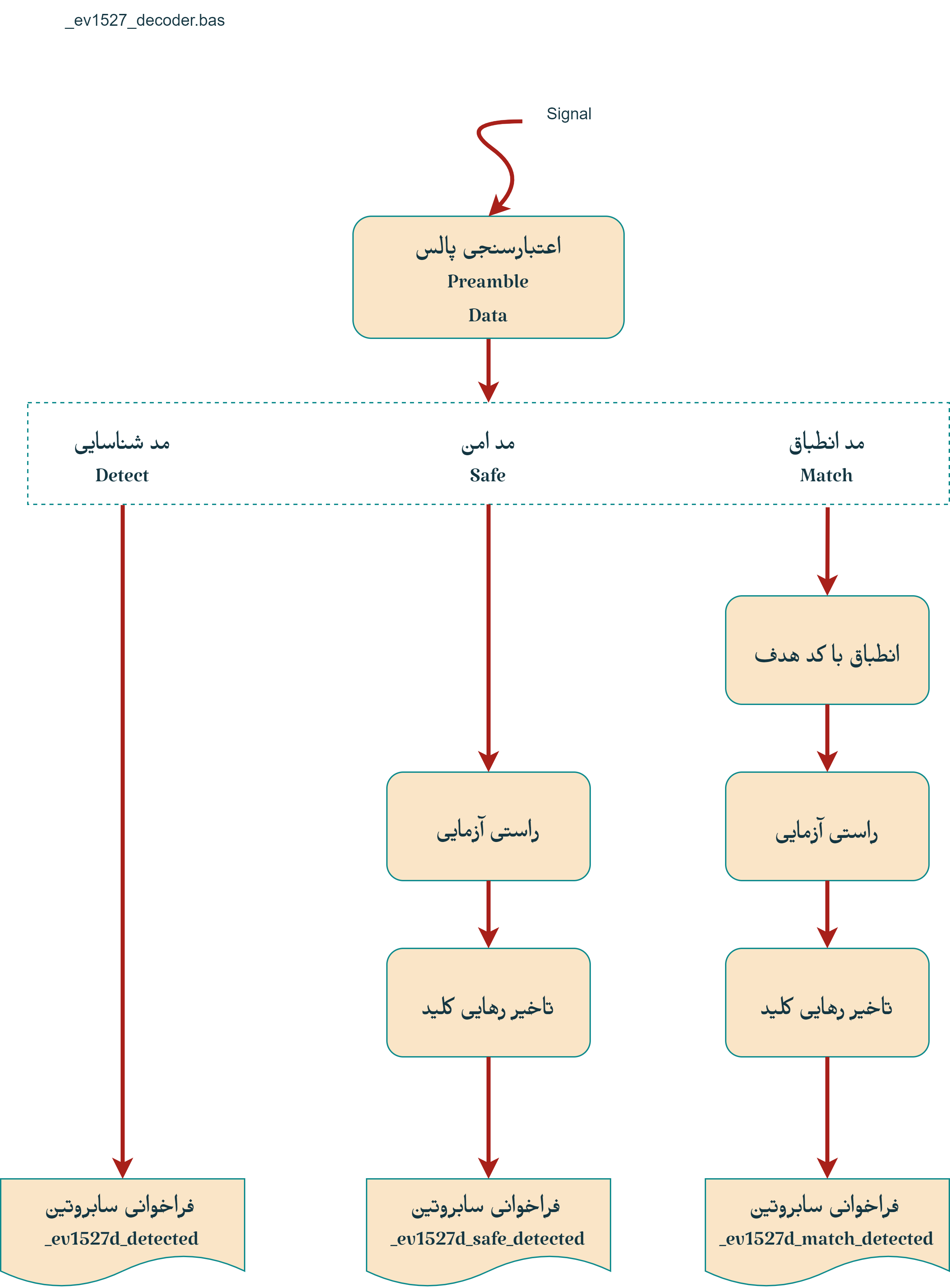
در این مد راستی آزمایی و تاخیر رها سازی کلید نیز اعمال شده و بعد از گذشتن از این دو فیلتر سابروتین safe\_detected فراخوانی و کد دریافتی از طریق متغیر Code قابل خواندن خواهد بود.

**مد تطبیق (match) :**

این مد زمانی استفاده میشود که ما منتظر کد خاصی هستیم. برای استفاده از این مد ابتدا آرایه 3 بایتی target\_code رو مقدار دهی کنید سپس مد رو فعال کنید. در این حالت کد های منطبق بر کد هدف شناسایی شده سپس از فیلتر راستی آزمایی و تاخیر رهاسازی کلید گذشته و نهایتا سابروتین match\_detected فراخوانی و کد دریافتی از طریق آرایه 3 بایتی Code قابل خواندن خواهد بود.

لازم به ذکر است که در این مد فقط 20 بیت آدرس target\_code جهت تطبیق در نظر گرفته میشوند و مقدار 4 بیت دیتای آخر مهم نیست.

در هر زمانی می توانید از هر مد مستقیما به مد دیگر سوئیچ کنید



# **روش استفاده**

## اضافه کردن ماژول به برنامه:

ماژول رو ابتدای برنامه و بعد از تعریف regfile و crystal اضافه کنید

سه سابروتین detected, safe\_detected, match\_detected را به انتهای برنامه اضافه کنید

این سابروتین ها حتما باید موجود باشند در غیر اینصورت با خطای کامپایل مواجه میشود

کدی که داخل این سابروتین ها نوشته میشوند باید سریع باشد زیرا تا زمانی که این سابروتین ها به صورت کامل اجرا نشوند عملیات شناسایی دیتا فریم های بعدی آغاز نخواهد شد

## 

## مد شناسایی یا Detect:

برای فعال سازی ماژول در حالت شناسایی کد مطمئن شوید که وقفه ی سراسری فعال باشد سپس start\_detect\_mode را فراخوانی کنید

در این حالت پس از دریافت هر کد سابروتین detected فراخوانی شده و کد دریافتی از طریق آرایه code قابل خواندن است

متغیر کد یک آرایه 3 بایتی بوده و متشکل از 20 بیت آدرس و 4 بیت دیتا است

در این مد فقط تایمینگ ها بررسی شده و کد از فیلتر راستی آزمایی و تاخیر رهاسازی عبور نمیکند.

با توجه به اینکه کد دریافتی بدون هیچ فیلتری باعث فراخوانی این سابروتین میشود اگر کدی که داخل این سابروتین نوشته میشود به اندازه کافی سریع نباشد (در حد چند میکروثانیه) دیتا فریم بعدی از دست خواهد رفت بنابراین از نوشتن دستوراتی مثل print و wait در این سابروتین خودداری کنید

## 

## مد امن یا Safe:

برای فعال سازی ماژول در حالت اسکن کد مطمئن شوید که وقفه ی سراسری فعال باشد سپس سابروتین start\_safe\_mode را فراخوانی کنید

در این حالت پس از دریافت هر کد سابروتین safe\_detected فراخوانی شده و کد دریافتی از طریق آرایه code قابل خواندن است

متغیر کد یک آرایه 3 بایتی بوده و متشکل از 20 بیت آدرس و 4 بیت دیتا است

## 

## 

## 

## مد تطبیق یا Match:

برای فعال سازی ماژول در حالت تطبیق

ابتدا متغیر target\_code را که یک آرایه 3 بایتی بوده را با کد مورد نظر مقدار دهی کنید

هنگام مقایسه این متغیر فقط 20 بیت آدرس آن در نظر گرفته میشود و مقدار 4 بیت دیتای انتهایی تاثیری ندارد

مطمئن شوید که وقفه ی سراسری فعال باشد سپس سابروتین start\_match\_mode را فراخوانی کنید

در این حالت آدرس کد دریافتی با آدرس target\_code مقایسه می شود و در صورت تطبیق زیر روال match\_detected فراخوانی شده

و کد دریافتی از طریق متغیر code که یک آرایه 3 بایتی است قابل خواندن خواهد بود

## 

## 

## 

## متغیر خواندنی Status

در هر قسمتی از برنامه میتوانید وضعیت ماژول را بررسی کنید

مقدار status می تواند برابر با یکی از ثابت های stop , detect\_mode , safe\_mode , match\_mode باشد

# 

# **تنظیمات**

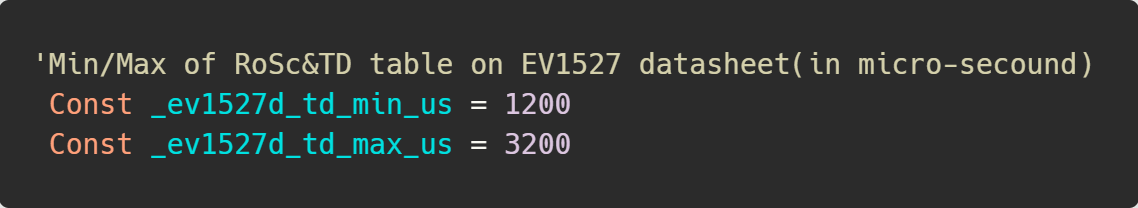
## بیشینه و کمینه طول پالس کامل دیتا:

این مقادیر داخل دیتاشیت ev1527 در قالب جدول RoSc&TD آورده شده اند

با توجه به غیر استاندارد بودن بعضی از ریموت ها و برای جلوگیری از عدم شناسایی این ریموت ها این بازه

کمی بیشتر در نظر گرفته شده

بقیه فیلتر ها مثل طول پالس preamble ، یک بودن و … بر اساس این مقادیر حساب خواهند شد



## درجه سخت گیری به تایمینگ ها:

مقدار این تابت تعیین میکند که کدام فیلتر ها بر سیگنال دریافتی اعمال شود و می تواند 1، 2 و یا 3 باشد

مقدار 1 فقط طول کامل پالس preamble و طول کامل پالس های دیتا را بررسی میکند

مقدار 2 علاوه بر فیلتر 1 طول یک بودن پالس preamble و پالس های دیتا را نیز بررسی میکند

و مقدار 3 علاوه بر فیتر 2 نسبت کل پالس preamble به یک بودن آن را نیز بررسی میکند که

بیشینه کمینه این فیلتر نیز قابل تنظیم است.

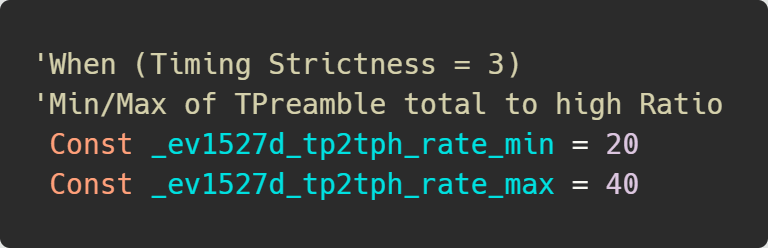
مقدار فیلتر هرچه بیشتر باشد چون احتمالا چند دیتا فریم نامعتبر شناخته میشود تا بالاخره یک دیتا فریم ایدآل دریافت شود ممکن است سرعت تشخیص را پایین بیاورد

## 

## بیشینه و کمینه نسبت کل پالس preamble به یک بودن آن:

این تنظیم فقط در صورتی که درجه سخت گیری روی 3 باشد استفاده میشود و بیشینه کمینه نسبت کل پالسpreamble به یک بودن آن را مشخص میکند

نسبت ایدآل 32 است اما با توجه به تست هایی که انجام شد به ندرت سیگنال های دریافتی نسبت ایدآل دارند و اکثرا نسبت 25 تا 35 دارند

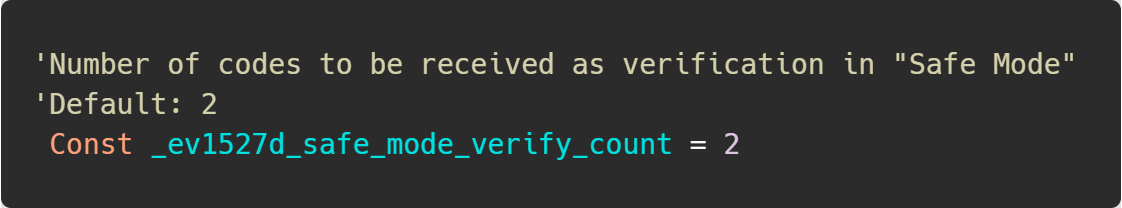


## تعداد راستی آزمایی در مد امن (safe):

تعداد کد هایی که بعد از دریافت کد اول باید دریافت شده و بر کد اول منطبق باشند تا کد دریافتی معتبر شناسایی شود و نهایتا تابع safe\_detected را فراخوانی کند

برای مثال اگر مقدار 2 باشد باید حتما 3 فریم با کد یکسان دریافت شود تا سابروتین safe\_detected فراخوانی شود

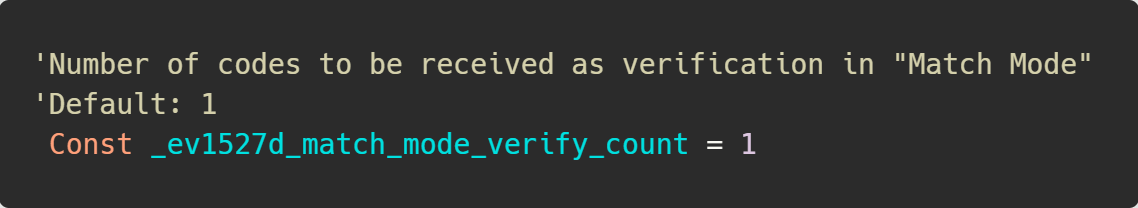
این تنظیم فقط در مد امن استفاده میشود



## تعداد راستی آزمایی در مد تطبیق (match):

تعداد کد هایی که بعد از دریافت کد هدف باید دریافت شده و بر کد هدف منطبق باشند تا کد دریافتی معتبر شناسایی شود و نهایتا تابع match\_detected را فراخوانی کند

برای مثال اگر مقدار 1 باشد باید حتما 2 فریم با کد یکسان دریافت شود تا تابع match\_detected فراخوانی شود



## تاخیر رهاسازی:

در هریک از مد های امن و تطبیق زمانی که کد معتبر دریافت شود برای اینکه زیر روال های مربوطه به صورت مداوم فراخوانی نشوند تاخیر رهاسازی در نظر گرفته شده به این صورت که سابروتین مورد نظر با دریافت کد معتبر یک بار فراخوانی می شوند و برای فراخوانی مجدد

باید کلید رها شده و همچنین زمان تاخیر رهاسازی هم بگذرد

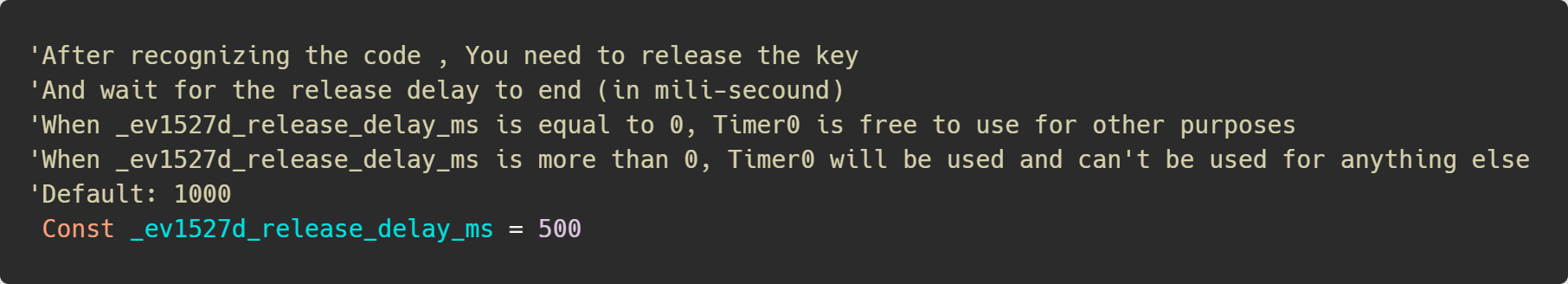
در این صورت با دریافت کد معتبر مجددا سابروتین های مربوطه فراخوانی میشوند

به عبارت دیگر با نگه داشتن یا فشردن و رها کردن پشت سر هم یک کلید سریع تر از این تاخیر تابع فقط یک بار فراخوانی میشود

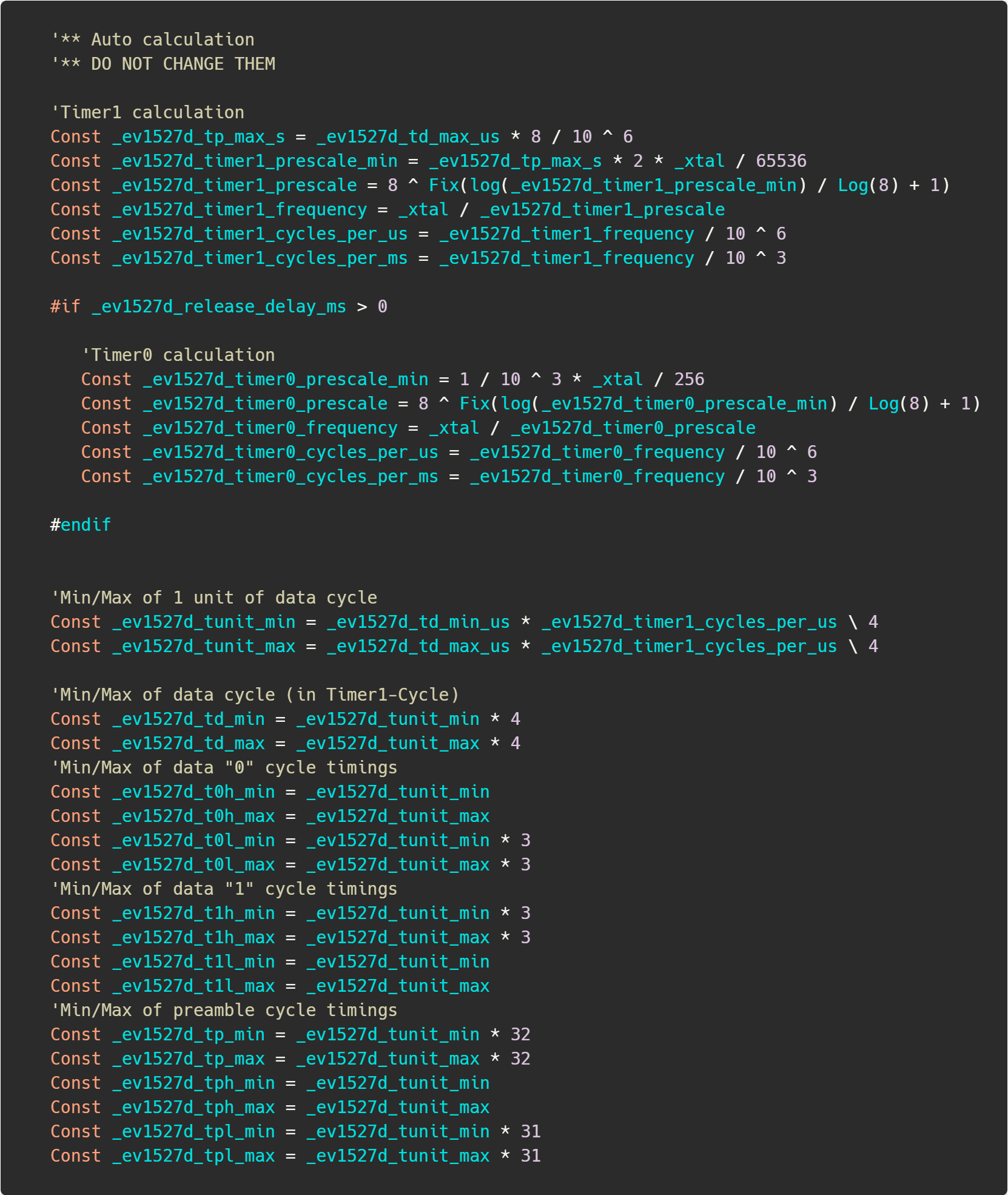
این تنظیم در مد های امن و تطبیق استفاده میشود و در مد تشخیص کارایی ندارد

در صورتی که این آپشن مقداری بیش از 0 داشته باشد تایمر 2 نیز درگیر و استفاده خواهد شد.

در صورتی که فقط از مد تشخیص استفاده میکنید این مقدار را صفر قرار دهید تا تایمر 2 آزاد شود



## این موارد اتوماتیک حساب میشوند و نیازی به تغییر ندارند

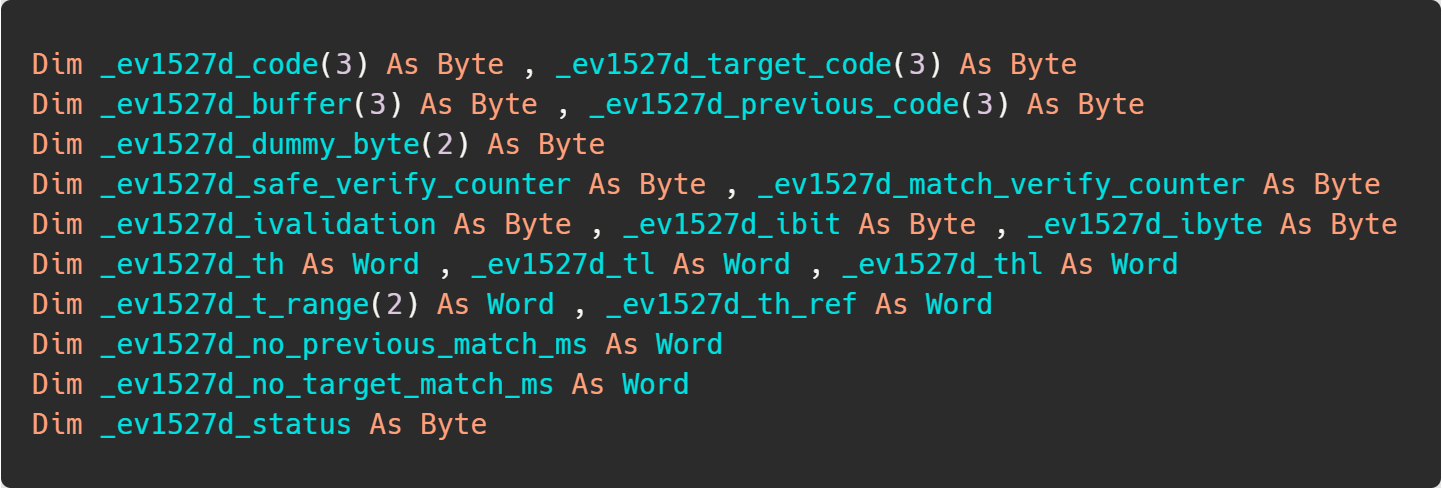


## 

## 

## متغیر های استفاده شده:

جمعا 36 بایت فضای SRAM استفاده خواهد شد



امکان اضافه کردن قابلیت های دیگر هم وجود دارد که بر حسب نیاز بتوان آن ها را فعال یا غیر فعال کرد

مثل:

* یادگیری چندین ریموت و انجام عملیات مقایسه و تطبیق با سرعت بالا
* ذخیره در ایپرام خارجی و داخلی
* سابروتین های رویداد های فشردن کلید و رها کردن کلید به صورت مجزا
* و ...

موفق باشید.