

**مدارهای منطقی**

**پروژه پایانی مدارهای منطقی : دستگاه ماشین خودکار**

**استاد درس : دکتر حاتم عبدلی**

دانشجو : محمد امین شهابی

نیمسال اول 1402-1403

مقدمه :

در این پروژه کا تلاش کردیم یک ماشین فروش خودکار بلیط برای مترو طراحی کنیم. در این پروژه از زبان VHDL استفاده کردیم.

**کتابخانه ها**:

کتابخانه **"library ieee;"** :

این کتابخانه به طور کلی به ما اجازه و دسترسی به توابع و ویژگی های مشخصی که توسط ieee مشخص میشود را میدهد.

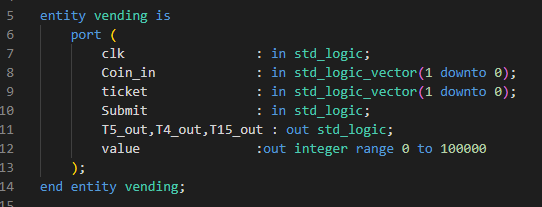
کتابخانه "**ieee.numeric\_std.all’;"** :

این کتابخانه برای انجام محاسبات عددی اضافه شده تا بتوانیم عملیات های مختلف مانند جمع تفریق و مقایسه رو انجام بدیم.

کتابخانه "**ieee.std\_logic\_1164.all’;"** :

این کتابخانه برای استفاده از داده های منطقی و تابع های مربوط به آنها استفاده شده است.

نتیجه: با استفاده از این دو کتابخانه در کد قادر به انجام عملیات های منطقی و عددی میشویم.

**entity**

در قسمت entity این برنامه ما یک ماژول به نام vending داریم که ورودی و خروجی های مختلف برنامه را تعیین میکنیم.

در این قسمت ما یک کلاک تعریف کردیم.

**Coin\_in** :

وظیفه Coin\_in گرفتن ورودی یا همان سکه است که یک وکتور در بازه 0 تا 1 است و تشخیص سکه ورودی بر عهده آن است به طوری که از 01 ورودیه Coin\_in باشد به معنیه سکه 500 تومانی و اگر 10 ورودیه Coin\_in باشد به معنی سکه هزار تومانی است.

**ticket** :

وضیفه این متغیر گرفتن نوع بلیط درخواستی از کاربر است که به ازای 00 بلیط 500 تومانی یکبار مصرف و 01 برای بلیط 1500 تومانی و 10 برای بلیط 4000 تومانی.11 هم don’t care است.

**Submit** :

این کلید به جای دکمه پایان برنامه استفاده میشود و زمانی که 1 باشه برنامه پایان میابد و تا زمانی که 0 باشه برنامه اجرا میشه.

**T”number”** :

این متغیر برای خروجی گرفتن تعریف شده و زمانی که هرکدام از متغیرهای

,T4(ticket 4000$),T15(ticket 1500$),T5(ticket 500$)درخواست داده شود و موجودی کافی باشد خروجی آنها یک میشود.

**Value** :

این یک متغیر از نوع integer است که وظیه نگهداریه موجودیه کاربر را دارد.موجودیه کاربر در بازه 0 تا 100000 تومان است و بیشتر از آن قبول نمیکند.

**architecture**

**signal temp** :

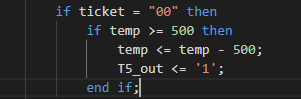
این سیگنال رو هم به عنوان موجودیه فعلی در نظر میگیریم و از نوع integer است و در بازه 0 تا 100000 هم بازه آن است.

**Process(clk)** :

این قسمت در شروع برنامه است و برای این است که شرط و مقدار دهی ها به صورت مکرر هر بار که clk میخوریم اعمال شود.

در مرحله اول process خروجیه بلیط هارو صفر میکنیم که برنامه هر بار تیکت های خروجی رو صفر کنه و برنامه قابلیت چند بار انجام شدن داشته باشه.

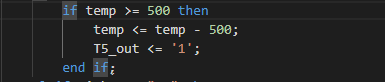
مقدار باقیمانده رو هم هربار صفر میکنیم که موجودی از پولی که وارد کردیم بیشتر نشه!

در بدنه Process چند شرط داریم که به صورت زیر می باشد :

**شرط اول اتمام عملیات درخواستی است که زمانی که کاربر دکمه Submit رو بزنه برنامه تمام خواهد شد.**

**وقتی که این شرط درست باشد :  
موجودیه فعلی درون موجودی وارد میشود و موجودیه فعلی صفر میشود.**

اصلی ترین قسمت برنامه شرط های اونه که بالا یک مدل از اونها اومده و به صورت زیر کار می کنه:

این شرط به این منظور است که تشخیص دهد نوع بلیط درخواستیه کاربر چیست.

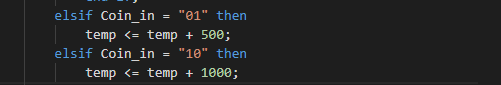
درون این یک شرط دیگر نیز وجود دارد که وظیفه آن این است که چک کند که temp یا همون موجودیه فعلیه کاربر از هزینه بلیط درخواستی بیشتر باشه.

شرط بالا درست باشد و وارد شرط بشیم دستگاه 500 تومان از موجودی کسر میکنه و خروجیه بلیط مورد نظر رو یک میکنه.

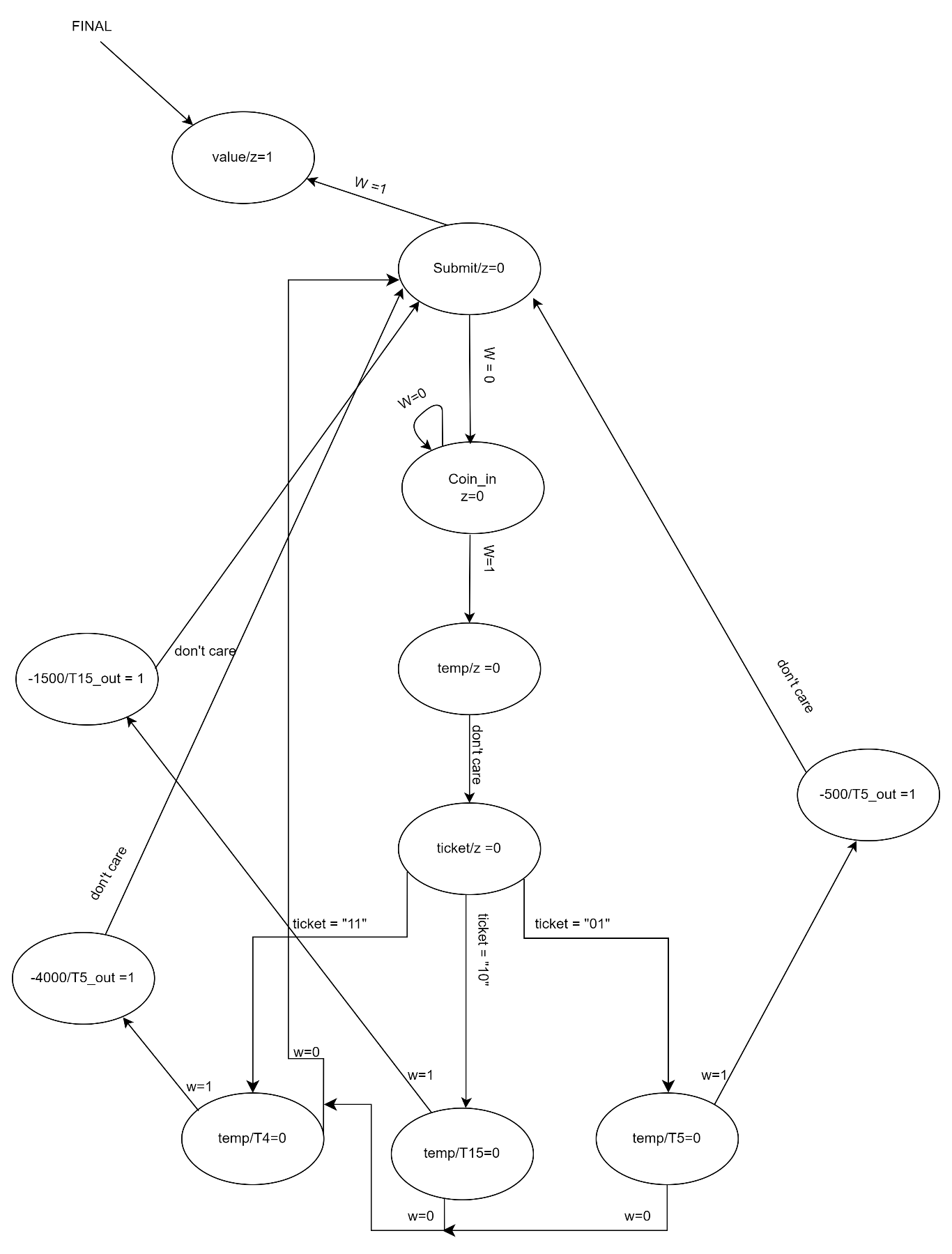
**توجه:**

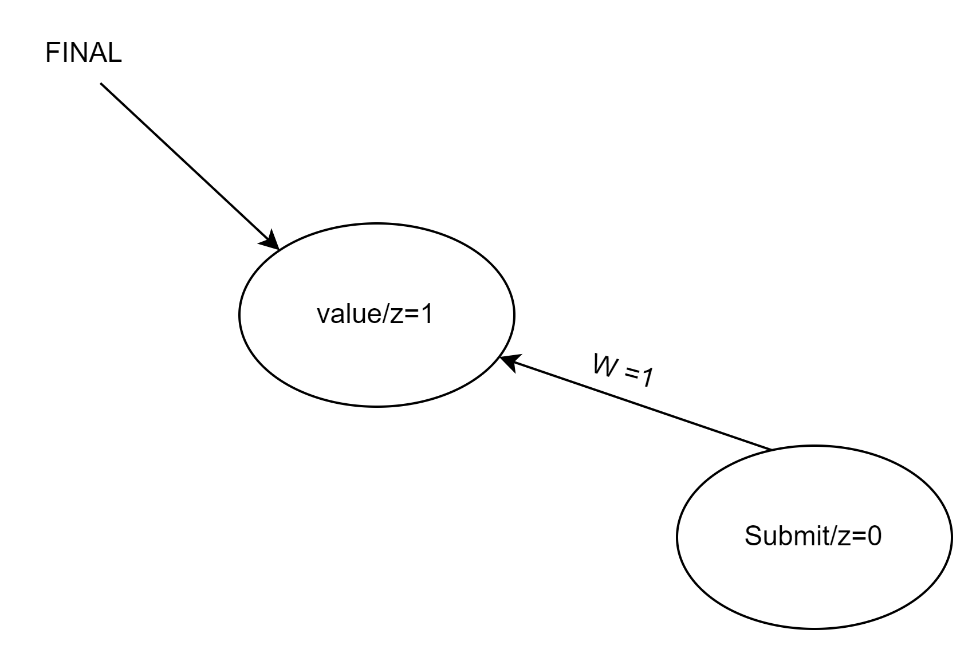
**میتونستیم هم شرط موجودی بزاریم به طوری که اگر موجودی منهای 500 بزرگتر از 0 بود بیا خروجی بده ولی چون نمیدونستم عدد منفی مشکل ایجاد میکنه یا نه این شرط رو اعمال نکردم!!!**

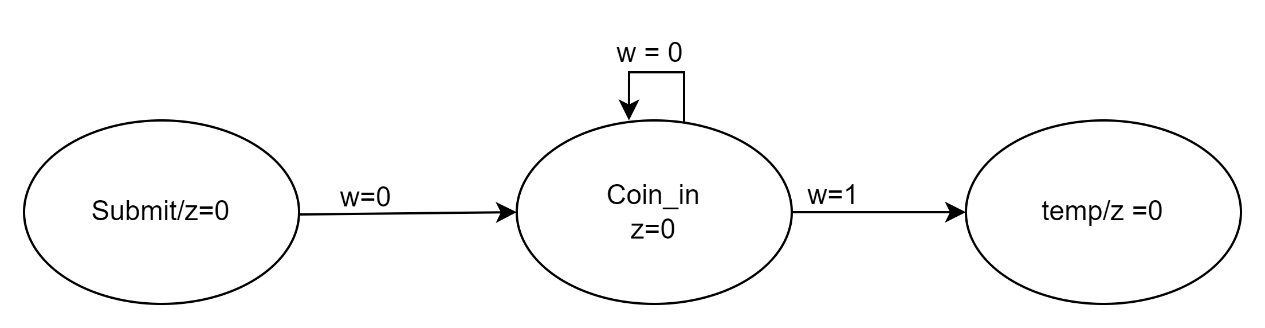
**!! این شرط برای تمام بلیط ها اعمال میشود !!**

**شرط بعدی که اعمال میشود شرط :**

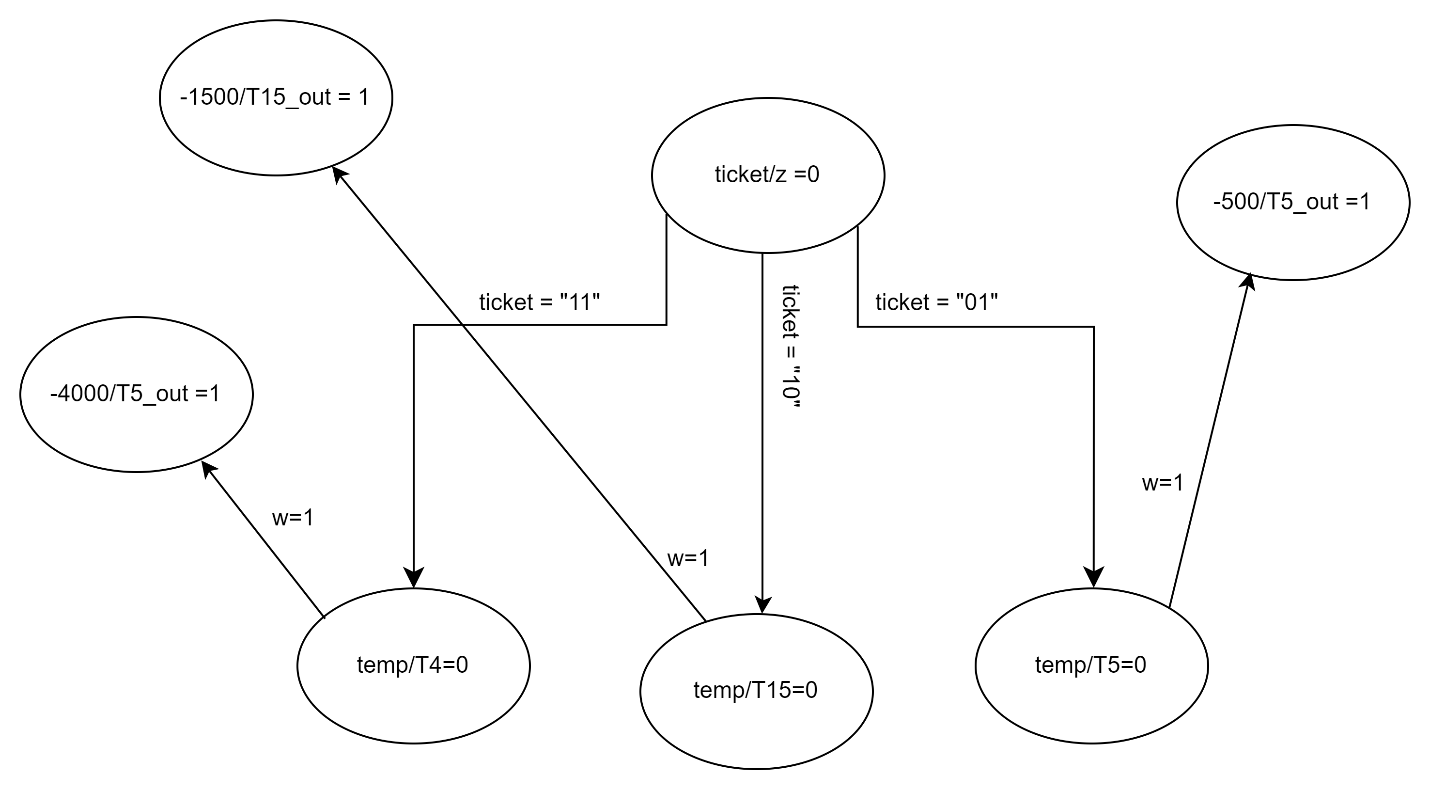
**فعال میشود که سکه هارا میگیرد و به موجودی اضافه میکند ! شرایط این عمر هم در بالا توضیح داده شده!**

 **FSM Chart** :

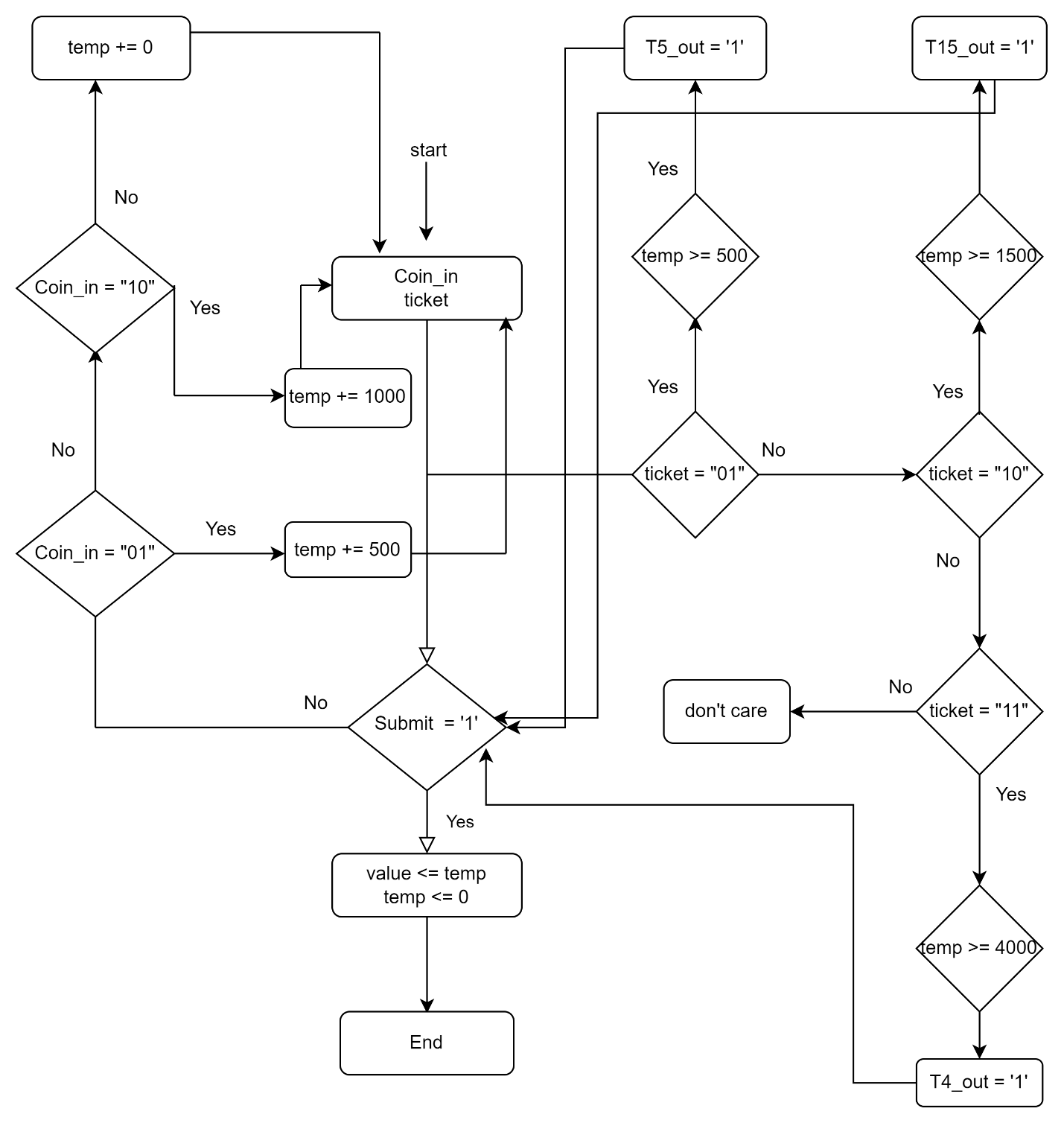
Fsm این مدار همانطور که در بالا میبینید رسم شده و تا زمانی که Submit = 1 نشود برنامه پایان نمیابد. 

هر بار که کاربر سکه ای وارد کند و درخواست بلیطی بدهد , مقدار سکه به temp اضافه میشود و وارد مرحله درخواست بلیط ها صادر میشود.

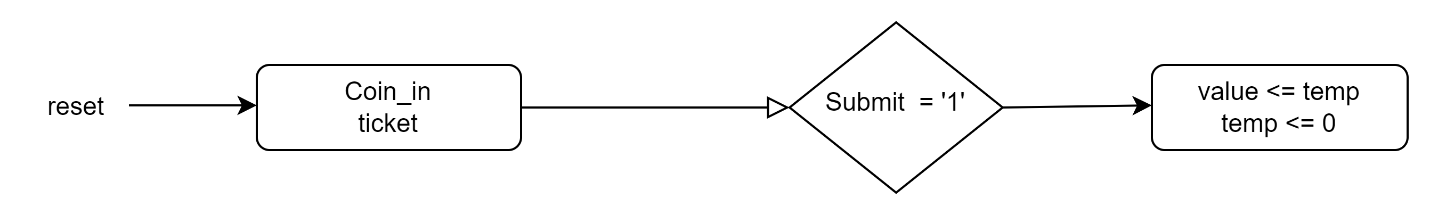
اگر temp از قیمت بلیط درخواستی بیشتر بود بلیط صادر میشود :



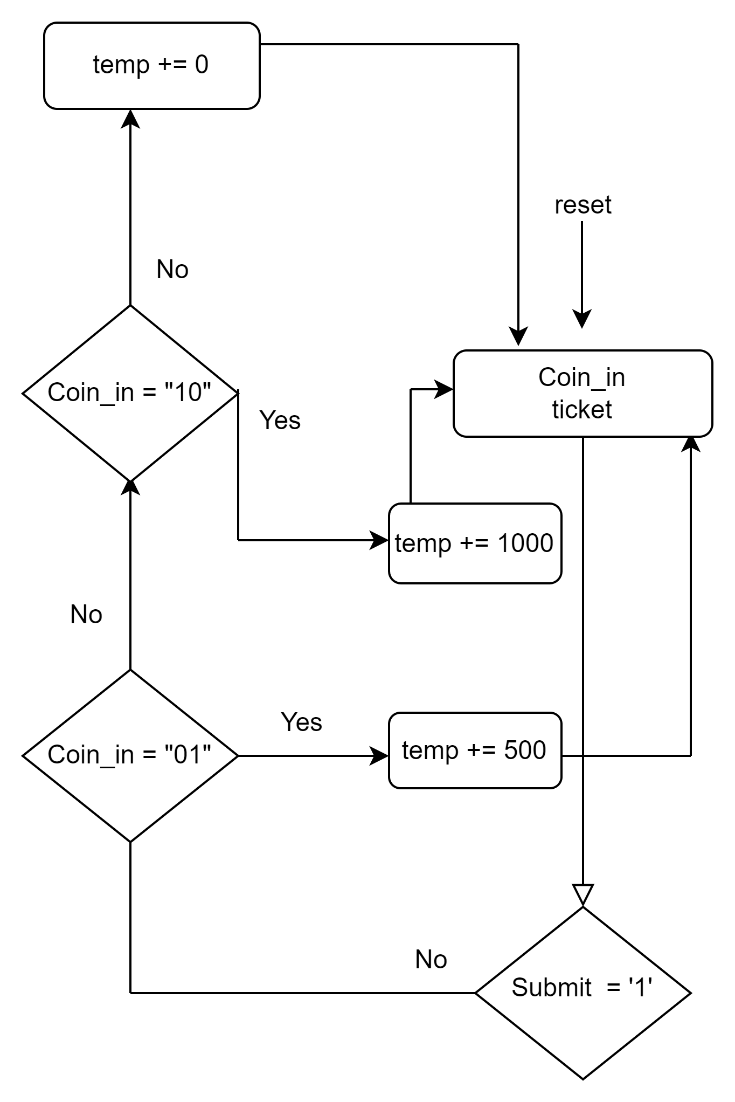
ASM Chart :



همانطور که در بالا مشاهده میکنید این فلوچارت تا زمانی که Submit = 0 باشد کار میکند.

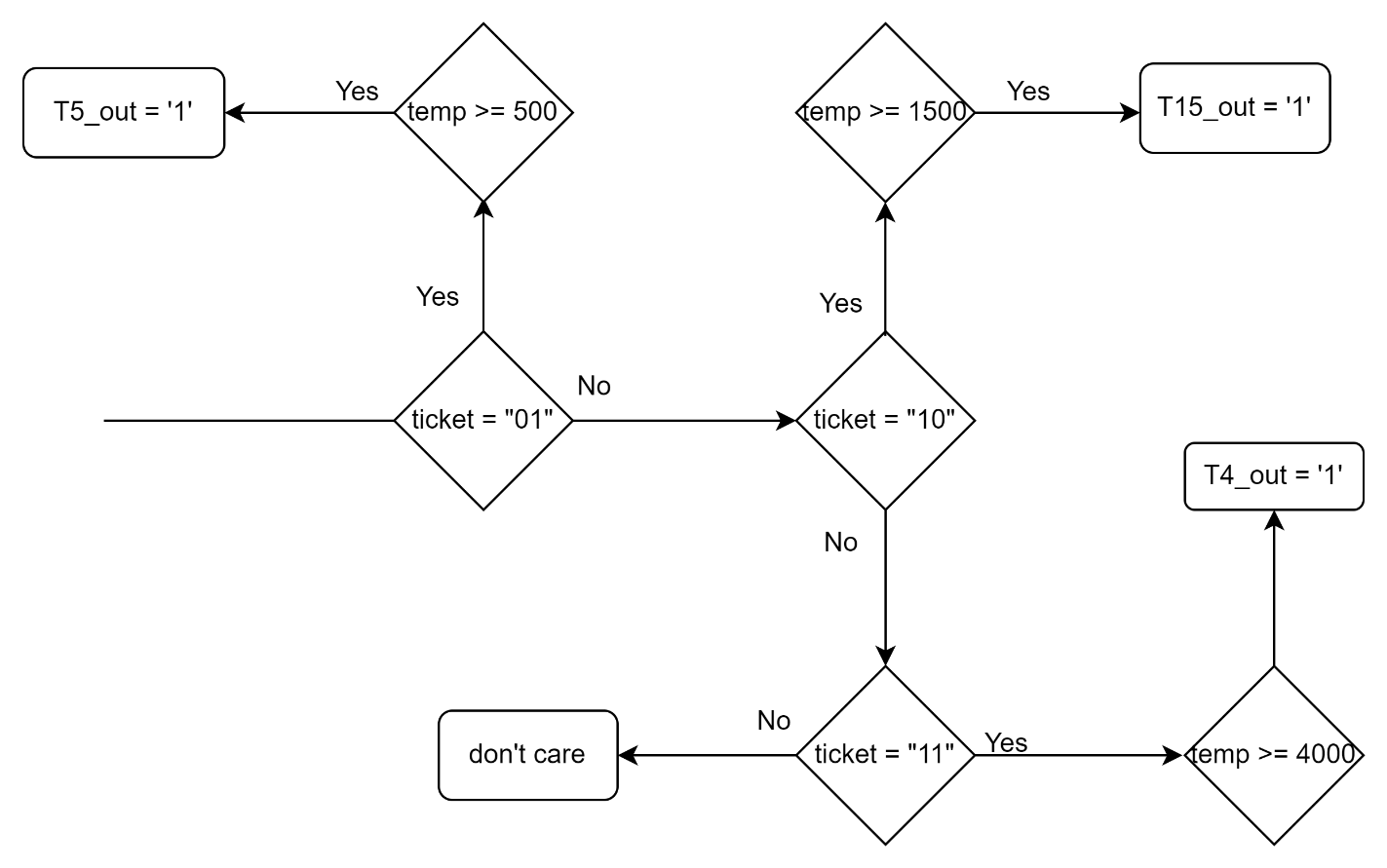
البته زمانی که Submit = 1 شود temp یا همان مقدار دارایی حال حاضر داخل value ریخته میشود و مقدار دارایی حال حاضر صفر میشود.(value همان باقی مانده پول کاربر است.)

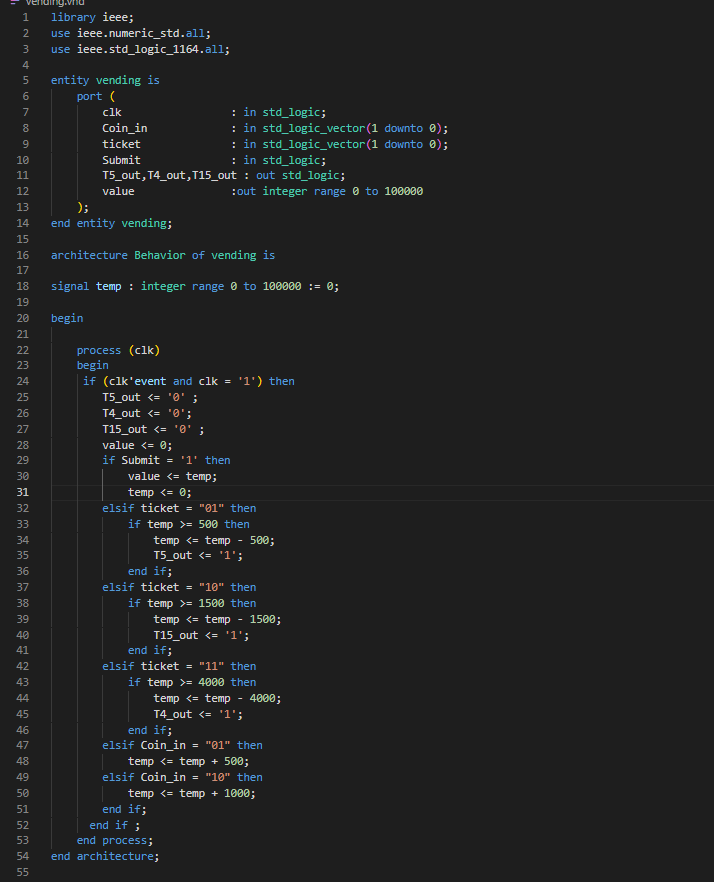
**While Submit = 0 :**



در این مرحله با غلط بودن شرط Submit سکه ها وارد شرط ها میشوند و هر جا که شرط مقدار سکه درست باشد داخل temp یا همان موجودی حال حاضر ریخته میشوند.

در مرحله بعد چک میشود که اگر بلیط درخواستی با موجودی حال حاضر مطابقت دارد خروجیه بلیط مورد نظر 1 شود و در غیر این صورت بلیطی تحویل داده نخواهد شد.



فرم نهایی کد :

زمانی که برای اولین بار کلاک میخورد و لبه بالا رونده را میبیند برنامه شروع به انجام کار میکند.