Unidade 3 - Plataformas clássicas de desenvolvimento de jogos

Unidade 3 - Plataformas clássicas de desenvolvimento de jogos

3.1. História
3.2. Características
3.3. Limitações
3.4. Exemplo: Atari 2600 (6507, TIA, RIOT)

Plataformas clássicas de desenvolvimento de jogos

História

Na década de 1970, cada máquina de jogo tinha um ou poucos jogos, já implementados e sem nenhum tipo de expansão. Muitas vezes implementados em hardware.

Um processo de produção diferente para cada "máquina" ou "jogo".

Processo de implementação: a cada novo jogo, todo o processo do zero!

Para o consumidor: comprar um novo hardware para cada jogo que fosse lançado? Muito caro.

Qual a solução: um hardware de propósito "mais geral", capaz de executar "programas de jogos" diferentes.

Plataformas clássicas de desenvolvimento de jogos

Características

Uma plataforma capaz de ser programada, com uma interface de entrada para este programa (cartucho, fita), e componentes de hardware que seriam usados por todos esses jogos.

A plataforma tem recursos específicos para permitir a programação dos jogos, como:

- capacidades gráficas para exibir imagens na tela;
- capacidades de produzir sons (não necessariamente);
- dispositivos de entrada "genéricos" para os jogadores;

Plataformas clássicas de desenvolvimento de jogos

Características

O programador deve conhecer a fundo esta arquitetura da máquina, para que possa usar corretamente esses seus recursos, pensados pelos seus designers.

Os jogos normalmente serão limitados por estes recursos, e visualmente, os jogos acabam tendo visuais parecidos, porque é justamente o que a plataforma oferece.

Até mesmo em arcades, a estratégia de "plataforma" foi utilizada (facilidade de programação e aproveitamento do hardware para outro jogo).

Plataformas clássicas de desenvolvimento de jogos

Limitações

Esse nível de conhecimento da arquitetura para se desenvolver um programa (um jogo) para ela era um limitador, pois portar um jogo poderia significar programar o jogo novamente.

Também por esta razão, era comum o mesmo jogo ter um visual muito diferente de uma plataforma para a outra, por não ter os mesmos recursos gráficos disponíveis. (Arcade x console).

Atari 2600 (originalmente Atari VCS)

Algumas versões: 1977, 1980, 1982, 2023

Atari 2600

Comandos no console

Atari 2600

Controles

Atari 2600


game program
video olympics
CX-2621
1-12 PONG-SUPERPONG
13-18 SOCCER
19-22 FOOZPONG
23-32 HOCKEY
33-34 QUADRAPONG
35-38 HANDBALL
39-42 VOLLEYBALL
43-50 BASKETBALL
Use with Standard Paddle Controllers
ATARI
game program
SKY DIVER
Use with Joystick Controllers
ATARI
CX2629
ATARI 2600
MARIO BROS.
ATARI 2600
MISSILE COMMAND
POLYVOX
ATARI
CARTUCHO ORIGINAL
ACTIVISION
POLYVOX
ATARI
JOUST
ATARI 2600 VIDEO GAME CARTRIDGE
SPRINT MASTER
ATARI 2600
10 GAMES IN 1

Atari 2600

Cartuchos

A mídia do atari foram os cartuchos. Resistente (até com proteção para não pegar poeira) e de fácil utilização.

Desde o início com um formato específico e também um visual característico, de "família" de produtos, porém, como vimos, com várias "fases" de design.

Ideia original: apenas a Atari lançaria cartuchos para a sua plataforma.

Realidade: outras empresas começaram a lançar cartuchos. A primeira delas foi a Activision.

Cada empresa lançou os cartuchos com o seu visual, a sua arte.

No Brasil as empresas relançavam os jogos, com ou sem os royalties pagos.

Atari 2600


ACTIVISION.
PRESENTS
STARMASTER™
Conceived and designed by Alan Miller
For use with the Atari® Video Computer System™
ONE PLAYER USING LEFT JOYSTICK CONTROLLER
Game 1:
Ensign
Game 3:
Wing Commander
Game 2:
Leader
Game 4:
StarMaster
ACTIVISION.
PRESENTS
KUNG-FU MASTER
For one or two players using Joystick Controllers
CRACKPOTS
cce
ATARI
CARTUCHO ORIGINAL
PRESSUR COOKER
ACTIVISION.
POLYVOX
SEAQUEST
Numa importante missão oceanográfica, seus mergulhadores são atacados por tubarões, que você tentará exterminar e apanhar seus homens-rãs. Cuidado pois um pequeno choque com os tubarões e você estará em pedaços, isso sem contar os submarinos que estarão disparando incessantes torpedos contra você. Não se esqueça do nível de oxigênio e quando soar o alarme, volte à superfície para reabastecer-se de oxigênio.
Lute com fibra e não saia desta todo molhado.
TABELA DE PONTOS
Inimigos a destruir
N.º de pontos
Submarino
20
Tubarão
20
CGC 51.485.036/0001-50 · Fabricado no Brasil
RIVER RAID
SUPERGAME
cce
C-811

Atari 2600


VENTURE
VIDEO GAME CARTRIDGE FOR USE WITH THE ATARI® VIDEO COMPUTER SYSTEM™ AND SEARS VIDEO ARCADE™
TAITO
Front Line
VIDEO GAME CARTRIDGE FOR USE WITH ATARI® 2600 VCS
MOUSE TRAP
VIDEO GAME CARTRIDGE FOR USE WITH THE ATARI® VIDEO COMPUTER SYSTEM™ AND SEARS VIDEO ARCADE™

Atari 2600


DACTAR
4 JOGOS
Chave nas Posições
RIVER RAID
ATLANTIS
DONKEY KONG
PAC-MAN
DACTAR
4 JOGOS
Chave nas Posições
ENDURO
SPACE INVADERS
DEFENDER
MEGAMANIA
DACTAR
4 JOGOS
A B C D
Chave nas Posições
NEXAR
FROGGER
FANTASTIC VOYAGER
DAMAS
DACTAR
2 JOGOS
Chaves nas Posições
MEGAMANIA
TENNIS
DACTAR
4 JOGOS
Chave nas Posições
MEGAMANIA
SEAQUEST
DEMON ATTACK
SUPERMAN

Atari 2600


BINGO
DRAGON TREASURE
Mr. POSTMAN
LASER GATE
ATLANTIS
COSMIC ARK
Q. BERT
JAW BREAKER
PAC MAN
BOBBY IS GOING HOME
AIR RAIDERS
TURMOILL
cce

Atari 2600


Mr. Postman
cce
C-801
DODGE 'EM
cce
C-842
DACTAR
4 JOGOS
A B C D
PAC - MAN
CHOPPER COMMAND
PIT
ENTOMBED
ATARI
Raiders of the Lost Ark
M·A·S·H
Games of the Century
DONKEY KONG
Nintendo
FROGGER
FROGGER
IMAGIC
DEMON ATTACK

Atari 2600


DONKEY KONG
by Nintendo
PLAYS LIKE THE REAL ARCADE GAME
VIDEO CARTRIDGE FOR USE WITH THE ATARI VIDEO COMPUTER SYSTEM AND SEARS VIDEO ARCADE
COLECO
Guide No. 78270B
DONKEY KONG
by Nintendo
CARTRIDGE INSTRUCTIONS
For Use With the Atari Video Computer System and Sears Video Arcade
• Adjusts to your skill level • For color or black-and-white TV
DONKEY KONG
by Nintendo
VIDEO GAME CARTRIDGE FOR USE WITH THE ATARI VIDEO COMPUTER SYSTEM AND SEARS VIDEO ARCADE


ATARI 2600 VIDEO GAME CARTRIDGE
DONKEY KONG
DONKEY KONG

Atari 2600

Cartuchos

A atari lançou jogos para o Atari 2600 de 1977 (Combat) até 1992 (Acid Drop, na Europa).

No entanto, outras empresas e até mesmo homebrews nunca pararam de serem lançados, principalmente pelo site AtariAge.com, recentemente comprado pela Atari.

No Brasil a produção de cartuchos de Atari foi de 1983 até 1992.

Atari 2600

Ainda tem relevância?

Retrogames estão na moda: pessoas querem jogar os jogos que jogavam na infância ou então novos jogadores querem descobrir e explorar jogos antigos.

Isso começou com os emuladores. Os primeiros que tiveram amplo uso foram criados em 1996 (stella, snes9x, massage) e 1997 (nesticle, mame).

O interesse foi só crescendo, e tais jogos começaram a ser lançados oficialmente em plataformas mais novas, a partir dos anos 2000.

Relançamentos de plataformas antigas também aconteceram.

Até se chegar a lançamentos de jogos inéditos para plataformas antigas.

Atari 2600

Ainda tem relevância?

É incrível, mas a resposta é sim!

É o console mais icônico!

Comércio de itens usados nunca parou, no mundo inteiro.

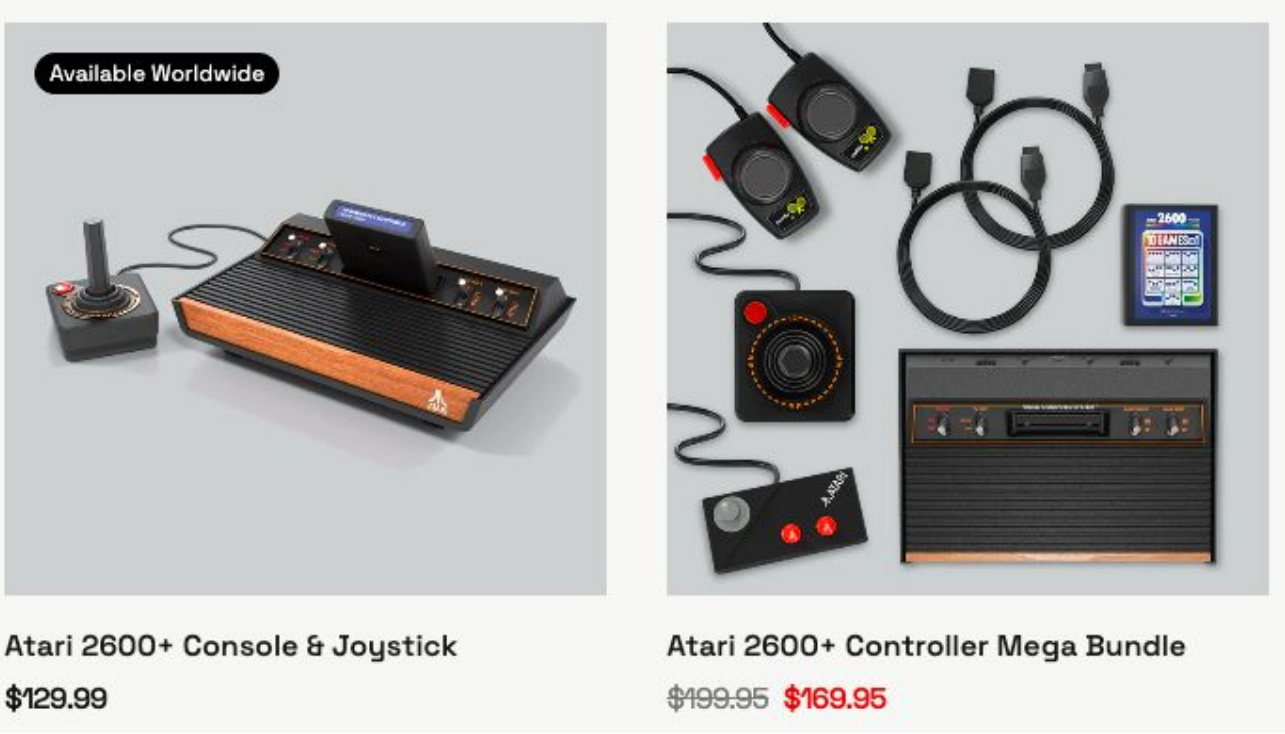
O desenvolvimento e discussão nunca parou, devido ao site Atariage.com. E o lançamento de "homebrews" por este site (recentemente comprado pela Atari).

Várias versões de Atari Flashback foram lançadas de 2004 até agora.

Em 2023 a Atari começou novamente a comercializar oficialmente produtos de Atari 2600 e Atari 7800.

Atari 2600

Comércio hoje (começou em 2023 no Atari.com)



Atari 2600

Comércio hoje

CX30+ Paddle Controller Bundle

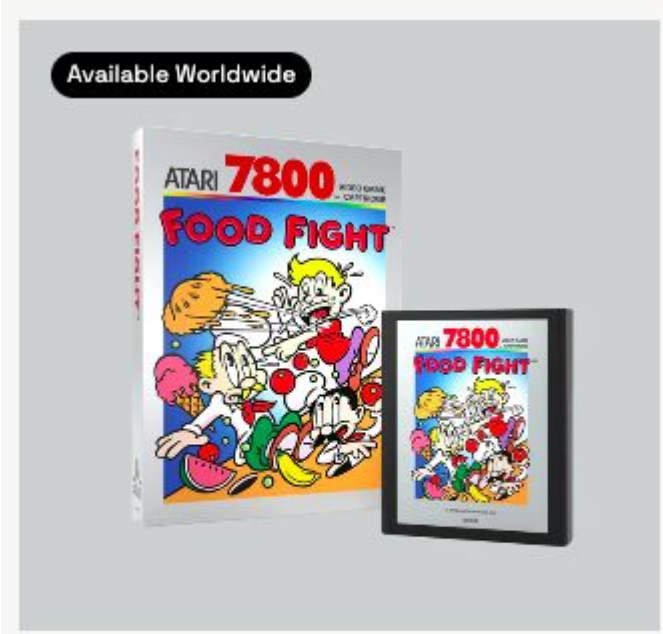
$39.99

Mr. Run And Jump 2600

$29.99

Atari 2600

Comércio hoje



Food Fight 7800

$29.99

Bentley Bear's Crystal Quest 7800

$29.99

Atari 2600

Comércio hoje

Adventure - Limited Edition

$99.99

Atari XP 50th Anniversary: Limited Edition Set Of 10

$999.99

Atari 2600

Comércio hoje - Atari Age

New 2600, 5200, 7800, Jaguar, and Lynx Games Arrive!

We've now made available all 21 new games released at 2023 Portland Retro Gaming Expo available for pre-ordering in our store! There are new games for the Atari 2600, 5200, 7800, Jaguar, and for the first time, Atari Lynx! We also have John Hancock's **Block'Em Sock'Em** for the Atari Jaguar in boxed form as well! There's an incredible variety of games in this collection of releases, and the talent of developers, artists, designers, musicians, and others working to make these games never ceases to amaze!

In addition to the games below, we also have added the **SNES2Atari Adapter** so it can be purchased separately (and we now sell three 7800 games that take advantage of it), an **AtariAge Baseball Cap**, and AtariAge/Atari PRGE 2023 t-shirts available in **black** and **white**. And, finally, we've made available our 2022 releases in cart/manual form with the box as an optional purchase, for those who aren't interested in owning the boxes.

Please keep in mind that we are opening these games for pre-ordering, so they will not ship right away. We are currently working on a large backlog of orders, and we will not ship these games until we get completely through that backlog. Opening these up for pre-orders allows us to gauge interest and order any additional materials we may need when it comes time to ship.

Below you can see the boxes for all the new games. Click on any of the images to jump directly to that game in the store:

Atari 2600

Comércio hoje - Brasil

Atari 2600

Comércio hoje - Brasil

Jataí: The Bee (Edição de Luxo)

R$350,00

Atari 2600

Comércio hoje

Atari 2600

Arquitetura

Processador MOS 6507 (variação do 6502) de 1.19 MHz. Outras opções da época: motorola 6800 e intel 8080 (mais caros).

128 bytes de RAM (bytes!!).

Normalmente 4KB na ROM do cartucho.

Outros dois chips: RIOT (MOS 6332) e TIA.

Atari 2600

Arquitetura

Atari 2600

Arquitetura

Sistema Operacional?

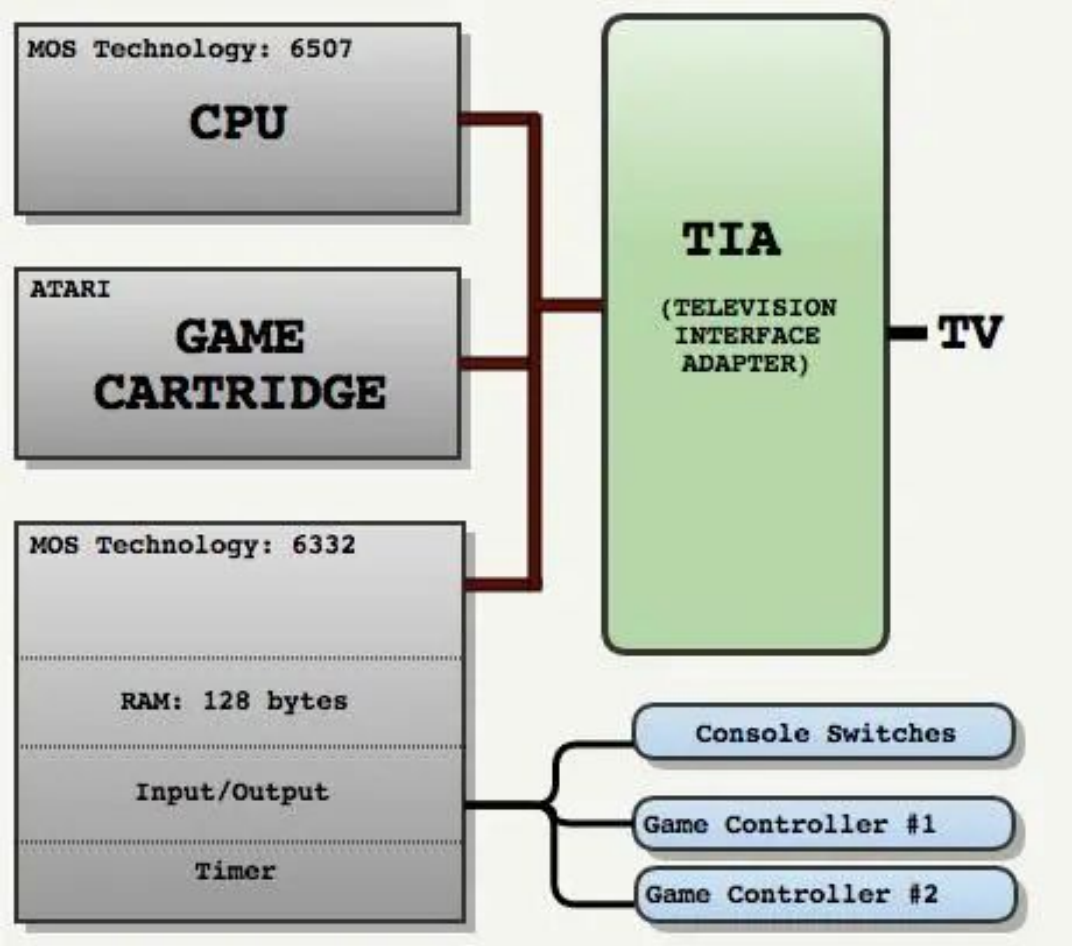
Bibliotecas?

Sistema de arquivos?

Compilador?

Game Engine?

Banco de Dados?



Atari 2600

Como funciona?

O processador MOS 6507 executa o programa que está em ROM (cartucho), usando a RAM, os dispositivos de entrada e a TV para exibir tudo, a partir do TIA e RIOT, cujos registradores são acessados por determinados endereços.

Desenvolvimento hoje: muito mais fácil que na época. Temos um bom assembler cross-platform, o DASM.

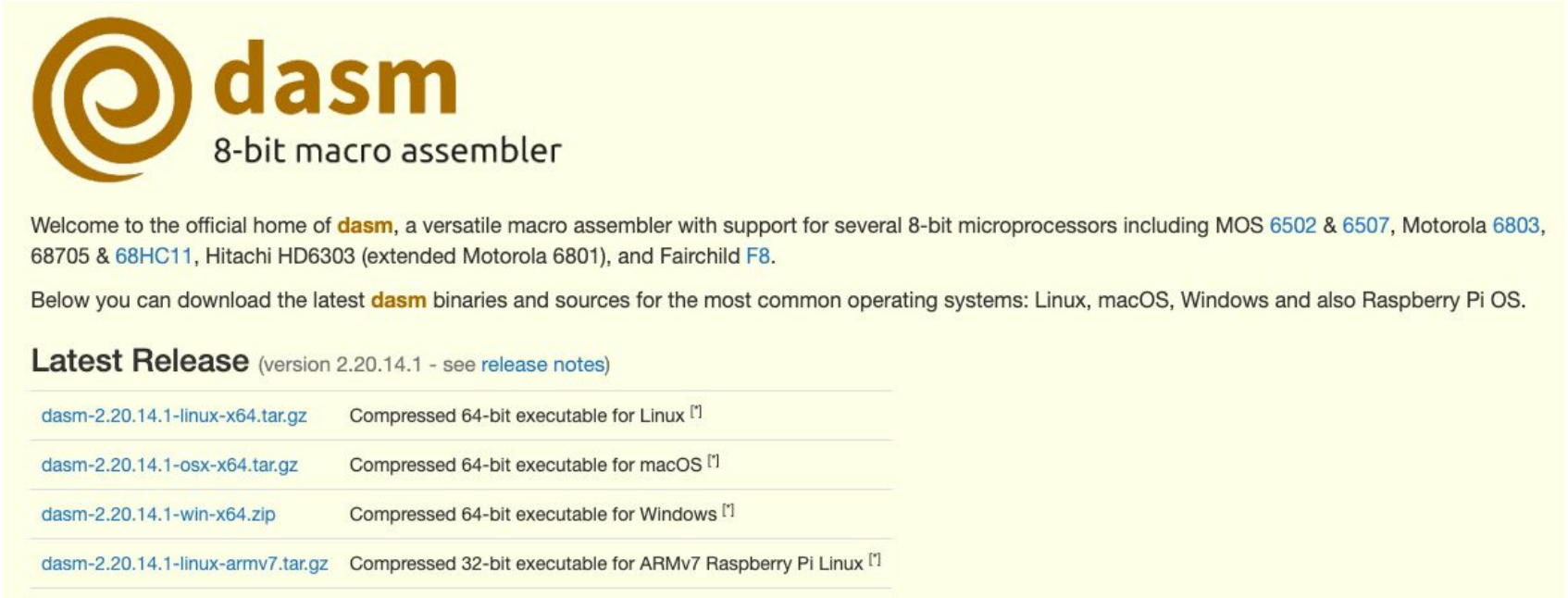
Temos bons emuladores, podendo testar modificações de forma quase instantânea!

Temos ferramentas auxiliares, como, por exemplo, para gerar o código os sprites.

Atari 2600

Ferramentas de desenvolvimento - assembler para 6507

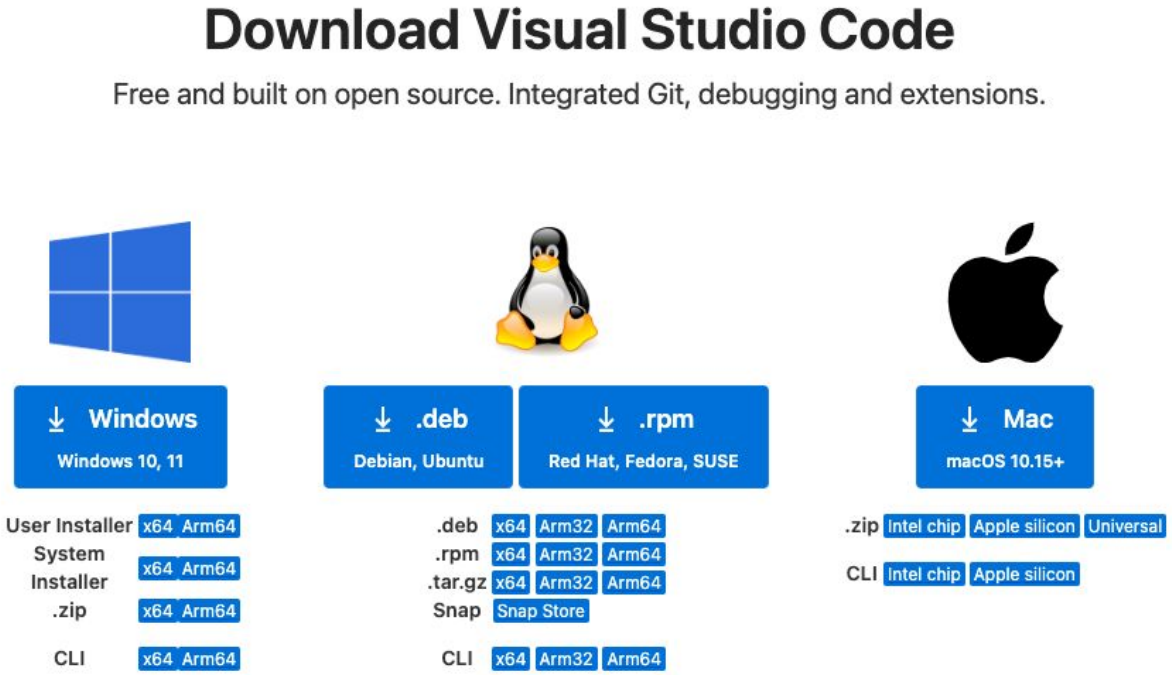
https://dasm-assembler.github.io



Atari 2600

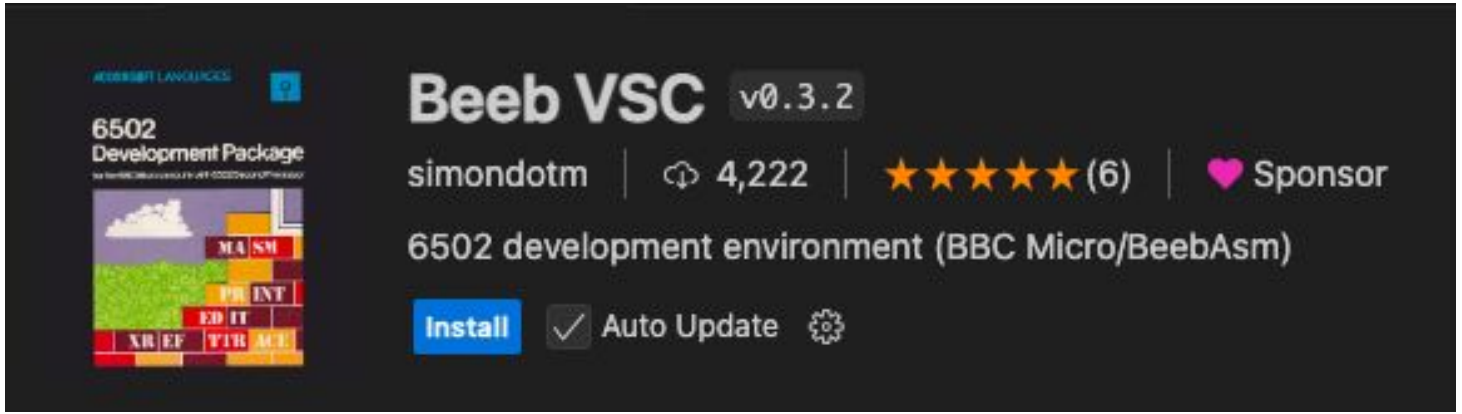
Ferramentas de desenvolvimento - editor de texto (ou outro de sua preferência)

https://code.visualstudio.com/download#



Atari 2600

Ferramentas de desenvolvimento - colorir sintaxe



Atari 2600

Ferramentas de desenvolvimento - emulador Stella

https://stella-emu.github.io/downloads.html

Latest Release (v/ 7.0)

Stella-7.0b-x64.exe	Binary installer (exe) for 64-bit Windows 7/8/10/11 (*)
Stella-7.0b-windows.zip	Binary ZIP for 64-bit Windows 7/8/10/11 (*)
Stella-7.0-macos.dmg	Binary DMG for macOS (ARM M1 and 64-bit Intel)
stella_7.0_amd64.deb	Binary 64-bit DEB for Ubuntu 22.04
stella-7.0-src.tar.xz	Source code tarball for all systems

Atari 2600

VSCode - Arquivos de configuração - tasks.json

```
{
    "version": "2.0.0",
    "tasks": [
        {
            "label": "Montar com DASM",
            "type": "shell",
            "command": "/Users/daniel/Documents/Atari/dasm/dasm",
            "args": [
                "${file}",
                "-f3",
                "-v0",
                "-I/Users/daniel/Documents/Atari/dasm",
                "-o${fileDirname}/${fileBasenameNoExtension}.bin"
            ],
            "group": {
                "kind": "build",
                "isDefault": true
            },
            "problemMatcher": []
        },
        {
            "label": "Executar no Stella",
            "type": "shell",
            "command":
"/Users/daniel/Documents/Stella/Stella/Contents/MacOS/Stella",
            "args": [
                "${fileDirname}/${fileBasenameNoExtension}.bin"
            ],
            "group": {
                "kind": "runTask",
                "isDefault": false
            },
            "problemMatcher": []
        }
    ]
}
```

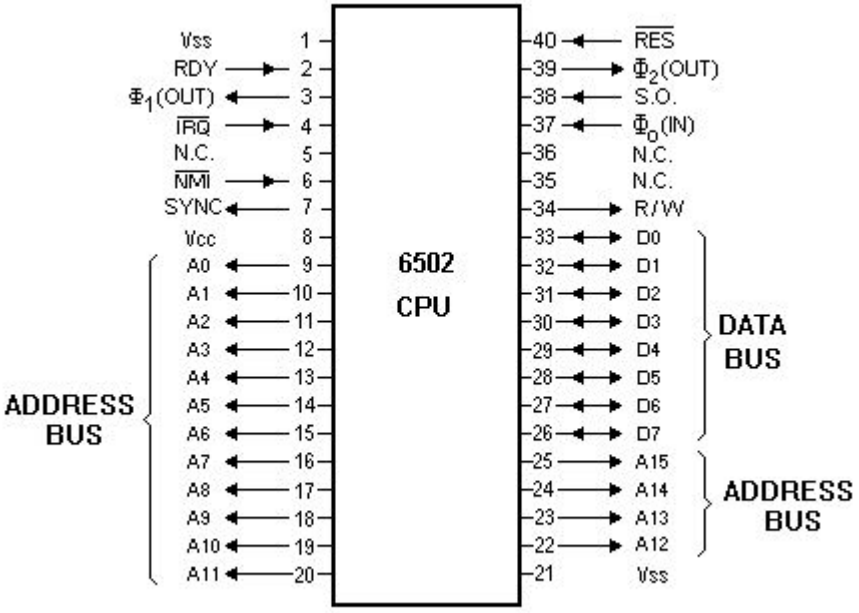
Atari 2600

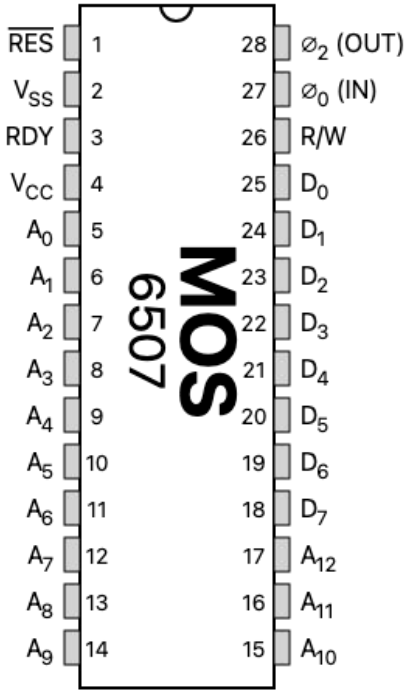
VSCode - Arquivos de configuração - keybindings.json

```
[
    {
        "key": "ctrl+alt+m",
        "command": "workbench.action.tasks.build",
        "when": "editorTextFocus"
    },
    {
        "key": "ctrl+alt+n",
        "command": "workbench.action.tasks.runTask",
        "args": "Executar no Stella"
    }
]
```

Atari 2600

MOS 6502 x MOS 6507





Atari 2600

MOS6507 - registradores

Registradores do MOS 6507

Registrador	Tamanho	Descrição
A (Acumulador)	1 byte	Usado para operações aritméticas e lógicas. É o principal registrador de dados.
X (Indexador X)	1 byte	Usado para indexação em operações de memória, contadores de laço e manipulação geral de dados.
Y (Indexador Y)	1 byte	Similar ao X, usado para indexação em instruções que suportam endereçamento indexado.
P (Registrador de Status)	1 byte	Contém flags que refletem o estado do processador, como resultados de operações e interrupções.
SP (Stack Pointer)	1 byte	Aponta para o topo da pilha na memória, localizada entre os endereços `0x0100` e `0x01FF`.
PC (Program Counter)	2 bytes	Aponta para o próximo endereço de instrução a ser executada na memória.

Atari 2600

MOS6507 - registradores

Acumulador (A):

- O registrador mais importante para cálculos.
- Operações aritméticas (como soma e subtração) e lógicas (como AND, OR e XOR) geralmente envolvem o acumulador.
- Exemplo: `ADC (Add with Carry)` adiciona um valor ao acumulador.

Atari 2600

MOS6507 - registradores

Registradores de Índice (X e Y):

- Usados principalmente para endereçamento indexado e manipulação de dados.
- Exemplo:
  - **X**: Pode ser usado como offset ao acessar arrays em memória.
  - **Y**: Similar ao X, mas também usado em algumas instruções de varredura de memória.
- Exemplo de instrução: `LDA $2000,X` carrega o valor do endereço `0x2000` + valor de X no acumulador.

Atari 2600

MOS6507 - registradores

Registrador Program Status (PS):

- Armazena flags que indicam o resultado de operações e o estado do processador.
- Cada bit (bit 7 até bit 0) no registrador de status possui um significado:
  - **N (Negative)**: Resultado negativo.
  - **V (Overflow)**: Overflow em operações aritméticas.
  - **B (Break Command)**: Indica interrupção por uma instrução BRK.
  - **D (Decimal Mode)**: Define se a aritmética usa o modo decimal (BCD).
  - **I (Interrupt Disable)**: Desativa interrupções IRQ. **Não faz sentido no 6507**.
  - **Z (Zero)**: Resultado igual a zero.
  - **C (Carry)**: Carry ou borrow em operações aritméticas.

Atari 2600

MOS6507 - registradores

Stack Pointer (SP):

- Aponta para o topo da pilha, que é uma região fixa na memória (`0x0100` a `0x01FF`).
- A pilha é usada para armazenar valores temporários, endereços de retorno e status durante chamadas de sub-rotinas ou interrupções.
- É manipulada automaticamente por instruções como `PHA` (Push Accumulator) e `PLA` (Pull Accumulator).

Atari 2600

MOS6507 - registradores

Program Counter (PC):

- Aponta para o próximo endereço na memória de onde será buscada uma instrução.
- Incrementado automaticamente após cada instrução, mas pode ser alterado por instruções de salto (`JMP`, `JSR`, etc.).

Atari 2600

MOS6507 - representação de números

#10 - decimal

$80 - hexadecimal

%00110101 - binário

Atari 2600

MOS6507 - representação de números

Usa complemento de dois para representar números positivos e negativos.

Liga os bits Carry e/ou overflow do PS.

Atari 2600

MOS6507 - algumas instruções

```
LDA   ; load register A (accumulator)
LDX   ; load register X
LDY   ; load register Y

STA   ; store register A (acumulador)
STX   ; store register X
STY   ; store register Y
```

Atari 2600

MOS6507 - algumas instruções

```
CLC   ; clears the carry flag
ADC   ; add to accumulator with carry

SEC   ; sets the carry flag
SBC   ; subtract from the accumulator with carry
```

Atari 2600

MOS6507 - algumas instruções

```
INC    ; increment memory
INX    ; increment register X
INY    ; increment register Y

DEC    ; decrement memory
DEX    ; decrement register X
DEY    ; decrement register Y
```

Obs.: pode setar os bits N e Z.

Atari 2600

MOS6507 - algumas instruções

```
JMP   ; incondicional jump

BCC   ; Branch on carry clear (bit C = 0)
BCS   ; Branch on carry set (bit C = 1)
BNE   ; Branch on not equal to zero (bit Z = 0)
BEQ   ; Branch on equal to zero (bit Z = 1)
BPL   ; Branch on plus (bit N = 0)
BMI   ; Branch on minus (bit N = 1)
BVC   ; Branch on overflow clear (bit V = 0)
BVS   ; Branch on overflow set (bit V = 1)
```

Atari 2600

MOS6507 - exemplo de loop

```
        ldx #10         ; x = 10
Loop:
        dex             ; x–
        bne Loop        ; repetir até x == 0
```

Atari 2600

MOS6507 - demais instruções

Consultar http://www.6502.org/tutorials/6502opcodes.html

ADC	AND	ASL	BCC	BCS	BEQ	BIT	BMI	BNE	BPL	BRK	BVC	BVS	CLC
CLD	CLI	CLV	CMP	CPX	CPY	DEC	DEX	DEY	EOR	INC	INX	INY	JMP
JSR	LDA	LDX	LDY	LSR	NOP	ORA	PHA	PHP	PLA	PLP	ROL	ROR	RTI
RTS	SBC	SEC	SED	SEI	STA	STX	STY	TAX	TAY	TSX	TXA	TXS	TYA

Atari 2600

MOS6507 - modos de endereçamento

lda #80

lda $80

Atari 2600

MOS6507 - modos de endereçamento

```
lda #80         ; immediate mode: load with decimal 80.

lda $80
```

Atari 2600

MOS6507 - modos de endereçamento

```
lda #80      ; immediate mode: load with decimal 80.

lda $80      ; zero page mode: load with the value inside the memory position $80.
```

Atari 2600

MOS6507 - modos de endereçamento

```
lda #80      ; immediate mode: load with decimal 80.

lda $80      ; zero page mode: load with the value inside the memory position $80.

lda #$80     ; immediate mode with hexadecimal 80;
```

Atari 2600

MOS6507 - modos de endereçamento

Modo de Endereçamento	Descrição	Tamanho (bytes)	Ciclos	Exemplo
Imediato	O dado está diretamente na instrução.	2	2	LDA #$10
Zeropage	Pega o conteúdo do endereço.	2	3	LDA $20
Zeropage, X	O conteúdo do endereço + X (indexa o endereço).	2	4	LDA $20,X
Zeropage, Y	O conteúdo do endereço + Y (indexa o endereço).	2	4	LDX $10,Y
Absoluto	Um endereço de 16 bits é fornecido diretamente na instrução, pegando o seu conteúdo	3	4	LDA $1234
Absoluto, X	O conteúdo do endereço + X (indexa o endereço).	3	4-5	LDA $1234,X
Absoluto, Y	O conteúdo do endereço + Y (indexa o endereço).	3	4-5	LDA $1234,Y
Indireto	Saltará para o endereço de 16 bits formado pelo byte que está neste endereço mais o byte seguinte.	3	5	JMP ($AABB)
Indireto, X	O endereço é lido a partir do endereço + X.	2	6	LDA ($20,X)
Indireto, Y	O endereço é lido a partir do endereço da instrução e aí é somado Y a ele.	2	5-6	LDA ($20),Y

Atari 2600

Mapa de memória

0000 - 002C - TIA (escrita)
0030 - 003D - TIA (leitura)
0080 - 00FF - RIOT (RAM)
0280 - 0297 - RIOT (I/O,Timer)
F000 - FFFF - Cartucho (ROM)

Atari 2600

Por que a ROM fica entre $F000 e $FFFF

O processador 6502/6507 usa os endereços $FFFC e $FFFD para armazenar o vetor de reset (o endereço inicial de execução após um reset).

Mas como acessar com apenas 13 bits? Com a técnica de "espelhamento":

Como o MOS 6507 ignora os bits mais altos (bits 13-15), ele repete o espaço de memória de 8 KB em intervalos ao longo do espaço lógico completo de 16 bits.

Isso significa que:

A ROM do cartucho, por exemplo, é decodificada para ocupar o espaço lógico $F000–$FFFF (mapeado fisicamente para $1000–$1FFF).

Na prática, $F000 acessa o mesmo conteúdo que $1000, $F001 acessa $1001, e assim por diante (conectada fisicamente desta forma). Isso é para manter a compatibilidade com o restante do projeto do MOS6502, que começa lendo os endereços finais para começar a execução.

Atari 2600

Mapa de memória

Os endereços dos registradores do TIA não precisam ser decorados.

Basta incluir o arquivo "vcs.h" e usar os nomes que estão definidos lá.

Temos também o arquivo "macro.h", com outras definições que podem ser importantes.

Atari 2600

TIA - exemplo de registrador (cor do fundo)

Cores (128) do TIA versão NTSC abaixo. Fonte:
https://en.wikipedia.org/wiki/List_of_video_game_console_palettes
Máximo de 4 cores por linha: background, playfield, player0 sprite e player1 sprite.

Hue	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Luma																
0, 1																
2, 3																
4, 5																
6, 7																
8, 9																
10, 11																
12, 13																
14, 15																

Atari 2600

TIA - exemplo de registrador (cor do fundo)

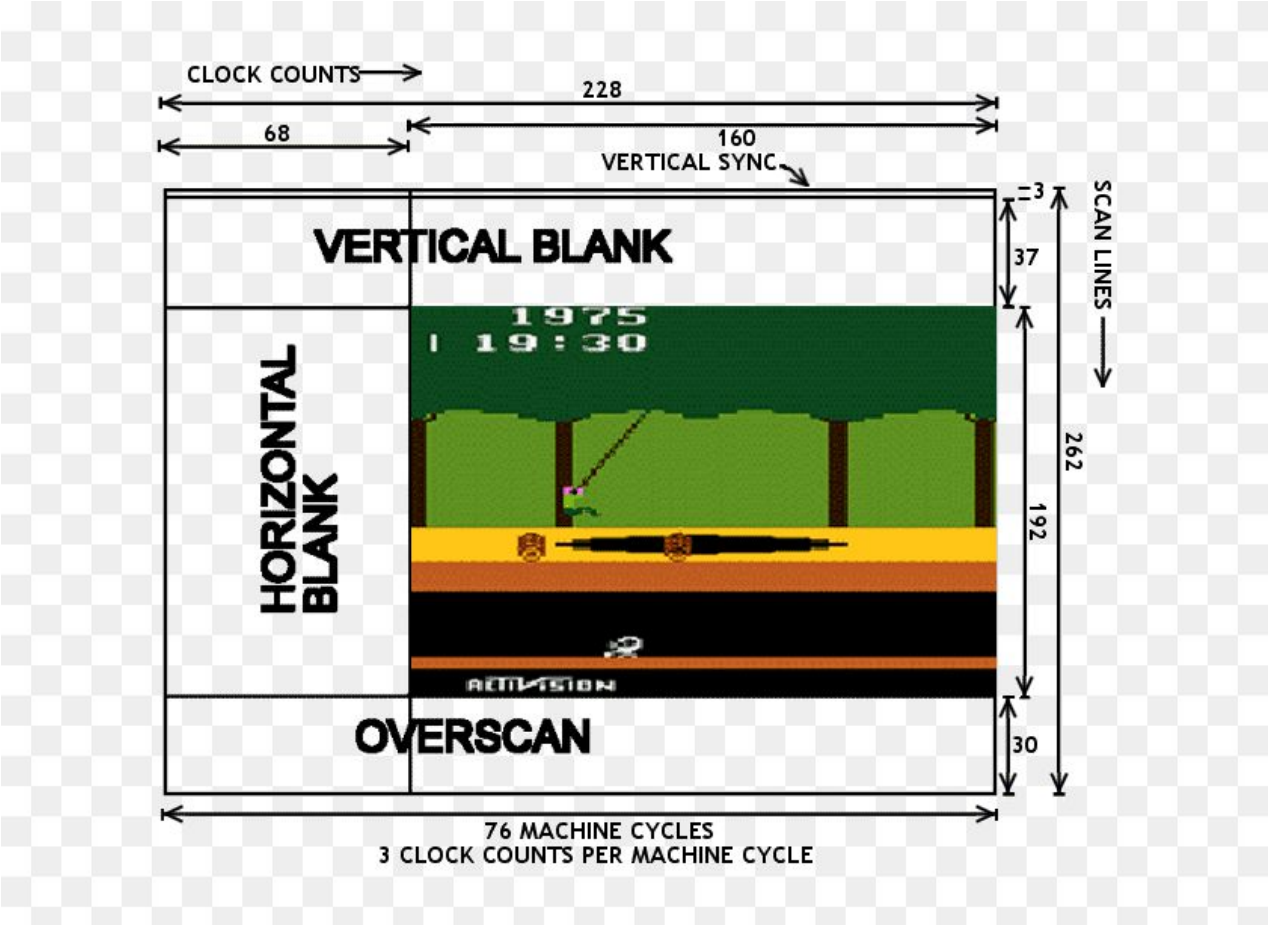
Mostrar arquivo **exemplo1-colorbg.asm**

Atari 2600

TIA - como desenha na tela?

Acompanha os raios disparados contra a tela na TV CRT.

Mostrar o **exemplo2-arcoiris.asm**



Atari 2600

TIA - duas cores no fundo, usando as scanlines

Mostrar arquivo **demo1_8.asm**

Atari 2600

TIA - objetos disponíveis

Background: cor de fundo.

Playfield: 20 blocos (cada um com 4 pixels), cobrindo a parte esquerda, podendo ser repetido ou refletido na parte da direita.

Player0 e Player1: 8 pixels por jogador (expansível).

Missile0 e Missile1: 1 pixel (projéteis para os jogadores)

Ball: 1 pixel (um elemento neutro, como a bola em Video Olimpics).

Atari 2600

TIA - Ball - registradores - demo2_1.asm

Registrador	Descrição
ENABL	Habilita ou desabilita a exibição da bola. Um valor de $02 ativa a bola; $00 desativa.
VDELBL	Controla o atraso vertical da bola. Se configurado como $01, o valor de ENABL será atrasado para o próximo scanline.
CTRLPF	Controla a largura da bola, além de outras propriedades do playfield e dos mísseis. Bits específicos (4-5) determinam se a bola será exibida com largura de 1, 2, 4 ou 8 pixels. $00 = 1px; $10 = 2px; $20 = 4px; $30 = 8px
RESBL	Reseta a posição horizontal da bola, alinhando-a com o contador horizontal do TIA. Usado para sincronizar a posição da bola com os elementos na tela. Strobe register, ou seja, basta gravar qualquer valor nele.
HMBL	Ajusta o movimento horizontal da bola. Define o deslocamento horizontal para o próximo quadro. Pode ser um valor positivo ou negativo para mover a bola à direita ou à esquerda. HMOVE deve ser escrito na próxima scanline.
COLUPF	Define a cor da bola (compartilhada com o playfield). Especifica o valor da cor em 4 bits (luminância e crominância).

Atari 2600

TIA - Players - registradores - demo2_2.asm

Registrador	Descrição
GRP0	Define o padrão gráfico do Player0. Cada bit de 8 bits representa um pixel na tela, com 1 ativando e 0 desativando o pixel. Ex.: $FF exibe todos os pixels ligados, enquanto $00 os apaga.
GRP1	Define o padrão gráfico do Player1, com funcionamento igual ao GRP0.
REFP0	Controla o espelhamento horizontal do gráfico do Player0. Valores: $00 para normal, $08 para espelhado.
REFP1	Controla o espelhamento horizontal do gráfico do Player1. Valores: $00 para normal, $08 para espelhado.
HMP0	Define o movimento horizontal do Player0. Valores em 4 bits (de $00 a $0F) controlam deslocamentos pequenos: $00 para nenhum deslocamento, $08 para mover 8 unidades para a direita, ou $F8 (valor negativo) para mover para a esquerda. HMOVE deve ser escrito em seguida.
HMP1	Define o movimento horizontal do Player1, com funcionamento igual ao HMP0. HMOVE deve ser escrito em seguida.
RESP0	Reseta a posição horizontal do Player0, alinhando-o ao contador horizontal do TIA. Não exige valores específicos; apenas escreva nele para sincronizar.
RESP1	Reseta a posição horizontal do Player1, com funcionamento igual ao RESP0.
COLUP0	Define a cor do Player0. Valores em 8 bits (crominância e luminância). Ex.: $F2 para laranja claro, $22 para azul escuro.
COLUP1	Define a cor do Player1, com funcionamento igual ao COLUP0.
NUSIZ0	Controla o tamanho do Player0. Valores: $00 para tamanho normal, $01 para duplicar largura, $03 para quadruplicar.
NUSIZ1	Controla o tamanho do Player1, com funcionamento igual ao SIZEP0.
VDELP0	Controla o atraso vertical do Player0. Valores: $00 para exibir normalmente, $01 para atrasar uma linha de varredura.
VDELP1	Controla o atraso vertical do Player1, com funcionamento igual ao VDELP0.

Atari 2600

TIA - registradores NUSIZx - demo2_3.asm

Bits (2-1-0)	Descrição
000	Uma cópia, largura normal (1 pixel).
001	Duas cópias, espaçamento próximo.
010	Duas cópias, espaçamento médio.
011	Três cópias, espaçamento próximo.
100	Duas cópias, espaçamento largo.
101	Uma cópia, largura dobrada (cada pixel é exibido com 2 pixels de largura).
110	Três cópias, espaçamento médio.
111	Uma cópia, largura quadruplicada (cada pixel é exibido com 4 pixels de largura).

Atari 2600

TIA - Missiles

Registrador	Descrição e Configuração
ENAM0	Habilita ou desabilita o "missile0". $00: Desabilitado. $02: Habilitado.
ENAM1	Habilita ou desabilita o "missile1". $00: Desabilitado. $02: Habilitado.
NUSIZ0	Controla o tamanho e número de cópias de "missile0". Exemplos: $00: Um míssil, 1x largura. $08: Duas cópias, 2x largura.
NUSIZ1	Controla o tamanho e número de cópias de "missile1". Exemplos: $00: Um míssil, 1x largura. $08: Duas cópias, 2x largura.

Atari 2600

TIA - Missiles - demo2_4.asm

Registrador	Descrição e Configuração
RESM0	Redefine a posição de "missile0" no próximo ciclo de escaneamento horizontal. Sem valores de configuração (apenas escrita).
RESM1	Redefine a posição de "missile1" no próximo ciclo de escaneamento horizontal. Sem valores de configuração (apenas escrita).
HMM0	Define o movimento horizontal de "missile0". $00: Sem movimento. Valores negativos ou positivos controlam a direção.
HMM1	Define o movimento horizontal de "missile1". $00: Sem movimento. Valores negativos ou positivos controlam a direção.
COLUP0	Define a cor do "missile0" (mesma cor do "player0"). $00-$FF: Código de cor em RGB compactado.
COLUP1	Define a cor do "missile1" (mesma cor do "player1"). $00-$FF: Código de cor em RGB compactado.

Atari 2600

TIA - Playfield

Registrador	Descrição e Configuração
PF0	Define os primeiros 4 bits do playfield (coluna mais à esquerda). Bits 4-7: Controlam os blocos. Exemplo: 00010000 (bit 4 ativo, os demais apagados).
PF1	Define os próximos 8 bits do playfield (coluna central esquerda). Bits 0-7: Cada bit controla um bloco no playfield. Exemplo: 10101010 (alternando blocos ativos e inativos).
PF2	Define os últimos 8 bits do playfield (coluna central direita). Bits 0-7: Cada bit controla um bloco no playfield. Exemplo: 11110000 (os 4 blocos mais à esquerda estão ativos).
CTRLPF	Controla o espelhamento e a prioridade do playfield. Bit 0: Controla o espelhamento horizontal do playfield (1 = espelhado). Bit 1: 0 (COLUPF) ou 1 (cores dos jogadores: player 0 na esquerda e player 1 na direita). Bit 2: Define a prioridade do playfield sobre os sprites (1 = prioridade ativada).
COLUPF	Define a cor do playfield e das balas. Bits 0-7: Código de cor compactado (afeta toda a área do playfield). Exemplo: 11100010 (um tom específico de cor).

Atari 2600

TIA - Playfield - demo2_5.asm

4 5 6 7	7 6 5 4 3 2 1 0	0 1 2 3 4 5 6 7
PF0	PF1	PF2

https://alienbill.com/2600/playfieldpal.html

Atari 2600

TIA - Movimentação horizontal - demo3_2.asm

Atari 2600

TIA - Movimentação vertical e tabelas de sprites - demo3_3.asm

https://alienbill.com/2600/playerpalnext.html

Atari 2600

TIA - Joystick - demo3_4.asm

Registrador	Descrição e Configuração
SWACNT	colocar o valor $00 para ler os joysticks
SWCHA	ler direções dos joysticks. 0 se pressionado, 1 se não está pressionado
INPT4	bit 7 é o botão do joystick 1: 0 se pressionado, 1 se não está pressionado
INPT5	bit 7 é o botão do joystick 2: 0 se pressionado, 1 se não está pressionado

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Right	Left	Down	Up	Right	Left	Down	Up
Joystick 1				Joystick 2			

Atari 2600

TIA - Shooting stars - demo3_5.asm

Atari 2600

TIA - Collision registers - demo3_6.asm

Registrador	Descrição
CXCLR	Tem que ser escrito com qualquer valor para voltar a ser sobrescrito.
CXM0P	Colisão entre Míssil 0 e Player 1 (bit 7 set) e Player 0 (bit 6 set)
CXM1P	Colisão entre Míssil 1 e Player 0 (bit 7 set) e Player 1 (bit 6 set)
CXP0FB	Colisão entre Player 0 e Playfield (bit 7 set) e Ball (bit 6 set)
CXP1FB	Colisão entre Player 1 e Playfield (bit 7 set) e Ball (bit 6 set)
CXM0FB	Colisão entre Míssil 0 e Playfield (bit 7 set) e Ball (bit 6 set)
CXM1FB	Colisão entre Míssil 1 e Playfield (bit 7 set) e Ball (bit 6 set)
CXBLPF	Colisão entre Bola e Playfield (bit 7 set)
CXPPMM	Colisão entre Players (bit 7 set) e Mísseis (bit 6 set)

Atari 2600

TIA - Chaves do console - demo3_7.asm (implementando uma pausa com select)

Registrador	Descrição
SWBCNT	colocar o valor $00 para ler as chaves
SWCHB	0 se pressionado, 1 se não está pressionado
Reset	bit 0: 0 = pressionado; 1 = não pressionado.
Select	bit 1: 0 = pressionado; 1 = não pressionado.
Color	bit 3: 0 = preto e branco; 1 = colorido.
Dificuldade 1	bit 6: 0 = Beginner (B); 1 = Advanced (A).
Dificuldade 2	bit 7: 0 = Beginner (B); 1 = Advanced (A).

Atari 2600

TIA - Som - demos do capítulo 4

Registrador	Descrição	Detalhes
AUDC0	Controle do Canal 0	Define a forma de onda e outras características (0-15) do som do Canal 0.
AUDC1	Controle do Canal 1	Define a forma de onda e outras características (0-15) do som do Canal 1.
AUDF0	Frequência do Canal 0	Define a frequência (0-31) para o som do Canal 0.
AUDF1	Frequência do Canal 1	Define a frequência (0-31) para o som do Canal 1.
AUDV0	Volume do Canal 0	Define o volume (0-15) para o Canal 0.
AUDV1	Volume do Canal 1	Define o volume (0-15) para o Canal 1.

Atari 2600

TIA - Som - demos do capítulo 4

Number	Name
1	buzzy
2	Buzzy/ramble
3	flangy/wavering
4	pure
6	pure/buzzy
7	reedy/rumble
8	white noise
12	pure lower
14	electronic/rumble
15	electronic/rumble

Atari 2600

Invaders - demo do capítulo 5

Atari 2600

Mais materiais:

https://www.randomterrain.com/atari-2600-memories-tutorial-andrew-davie-01.html

https://forums.atariage.com/forum/50-atari-2600-programming/

Atari 2600

IDE:

https://8bitworkshop.com

Referências Bibliográficas

Fontes Utilizadas na Apresentação

- GUTIERREZ, Oscar Toledo. Programming Games for Atari 2600. 1. ed. Lulu.com, 2022.
- http://www.6502.org/tutorials/6502opcodes.html
- https://en.wikipedia.org/wiki/List_of_video_game_console_palettes