



**AGT**technologies  
LCD Displays

# Manual do Desenvolvedor – Módulo Display AGT IHM v2.9.7

Rev. 0

## **O que há de novo na versão V2.9.7**

- 1.** Correção no firmware nos componentes Caixa de Texto e Entrada de Teclado, pois na versão anterior tinham parado de funcionar.

### **Tabela de Revisões**

Rev 0	Inicial
-------	---------

# Sumário

1 Descrição Geral.....	4
2 Características Elétricas e de Operação .....	4
3 Mecânica das Telas .....	5
4 Protocolo de Comunicação .....	14
4.1 Comandos de Operação .....	15
4.1.1 Comando para controle do beep interno .....	15
4.1.2 Comando para pular uma tela .....	15
4.1.3 Comando de habilitação do touch .....	16
4.1.4 Comando para leitura de status do touch .....	16
4.1.5 Comando para leitura da posição atual do touch .....	16
4.1.6 Comando para sinalizar eventos .....	17
4.2 Comando de Dados.....	17
4.2.1 Comando de escrita de VP (numérico).....	17
4.2.2 Comando de leitura de VP (numérico) .....	17
4.2.3 Comando curva .....	18
4.2.4 Comando de calibração .....	18
4.2.5 Comando de iniciar animação de tela .....	18
4.2.6 Comando de iniciar animação de imagem .....	18
4.2.7 Comando para desenhar linha na tela .....	18
4.2.8 Comando para desenhar círculo na tela.....	18
4.2.9 Comando para desenhar um ponto na tela.....	19
4.2.10 Comando para limpar tela .....	19
4.2.11 Comando para desenhar um polígono.....	19
4.2.12 Comando para desenhar um retângulo.....	19
4.2.14 Comando para atualizar o relógio.....	20
4.2.15 Comando para leitura relógio .....	20
4.2.16 Modo de utilização do VP .....	21

## 1 Descrição Geral

O Módulo Display AGTE IHM é um conjunto de circuitos eletrônicos capazes de oferecer a solução de uma IHM de baixo custo e alta funcionalidade, através de displays TFT de 3.5", 4.3", 7" e 10". As telas são controladas por um microcontrolador ARM Dual Core, operando a 200 MHz, que por meio de arquivos de configuração e arquivos de imagens criados pelos Aplicativo AGT IHM Designer, apresenta uma dinâmica de funcionamento orientado a toques e comandos pela porta UART. Para aumento da capacidade de memória, é possível utilizar o microSD Card como memória de expansão.

## 2 Características Elétricas e de Operação

As principais características elétricas podem ser visualizadas a seguir.

### *Alimentação:*

- 5V/1A (Modelo de 3.5");
- 5V/1A (Modelo de 4.3");
- 5V/2A (Modelo de 7.0");
- Bateria CR2032 3V para manutenção do RTC.

### *Tipos de Touch:*

- Resistivo 4 fios (todos modelos);
- Capacitivo (modelos 4.3", 7").

### *Suporte a cores:*

- Tela de 3.5", 4.3": 24 bpp;
- Tela de 7.0": 16 bpp.

### *Memória:*

- NAND Flash: 2 GB;
- SDRAM: 256 MB;
- Externa: até 8 GB (microSDCard).

### *Interfaces / Comunicação*

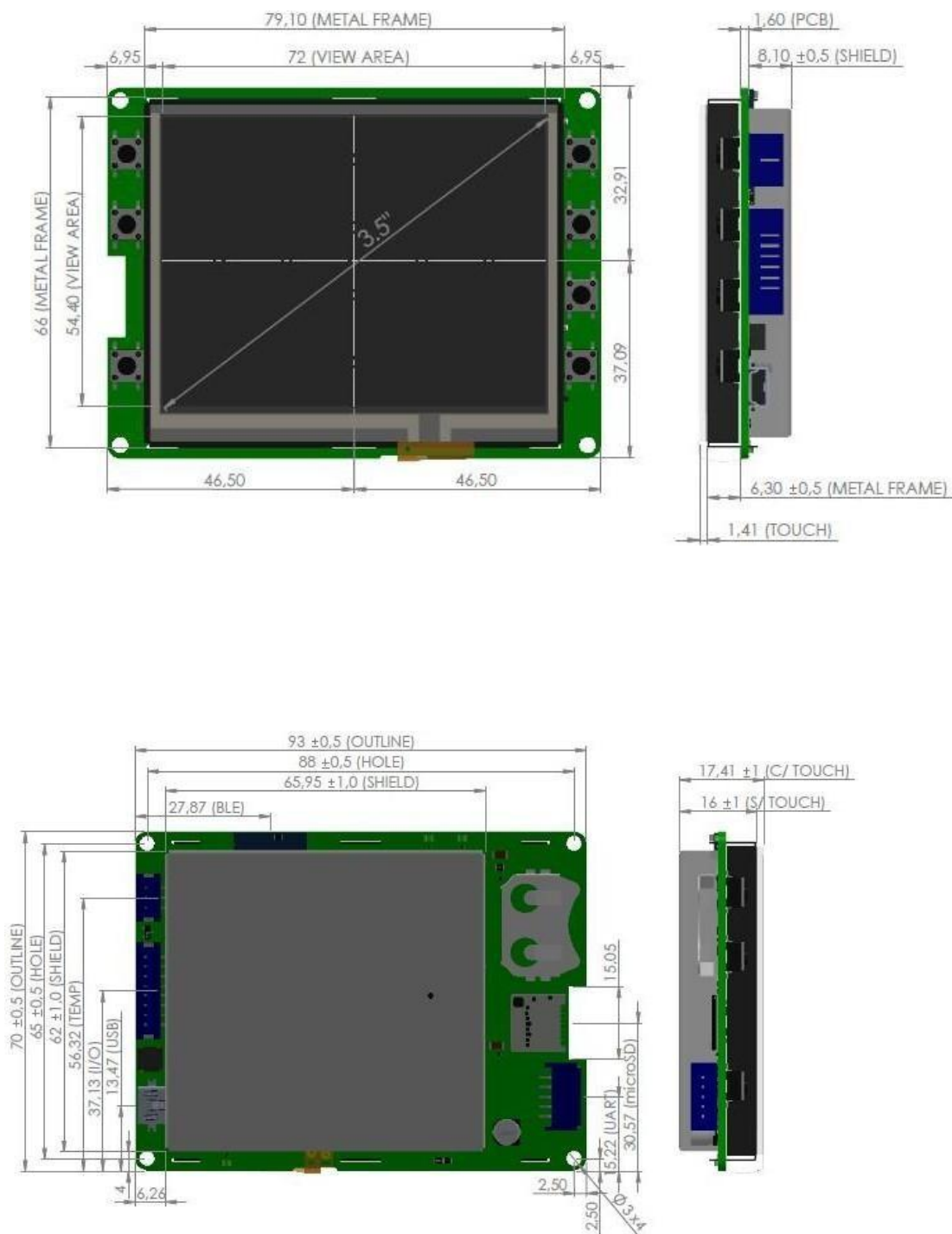
- UART TTL 5V (TX e RX) com conector FPC 8 pinos (mesma pinagem do DGUS);
- USB VCP (Virtual Communication Port = Porta Serial Virtual) com conector mini-B;
- microSD Card de até 8 GB;

### *Temperatura de operação:*

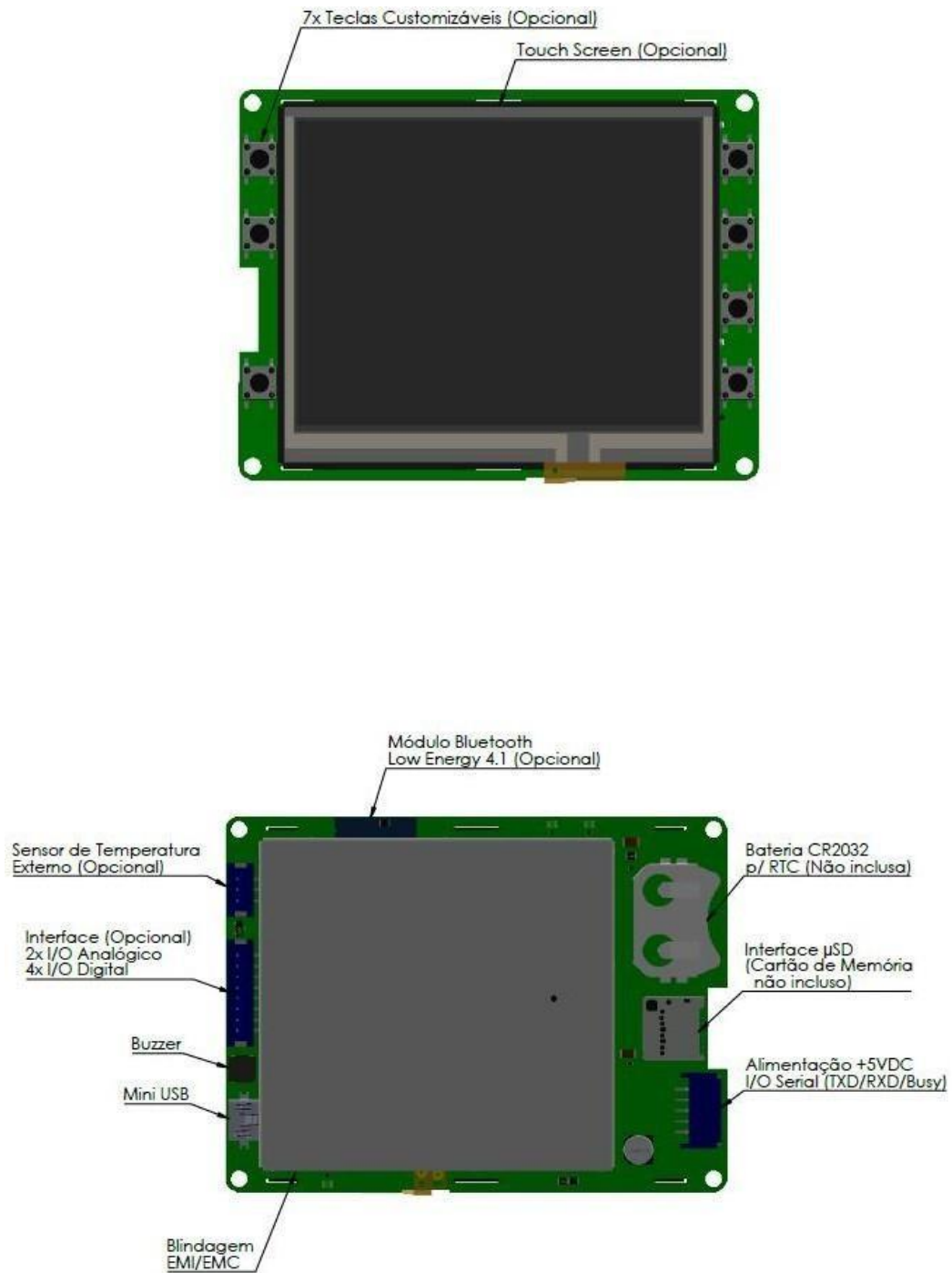
-20°C a +70°

### 3 Mecânica das Telas

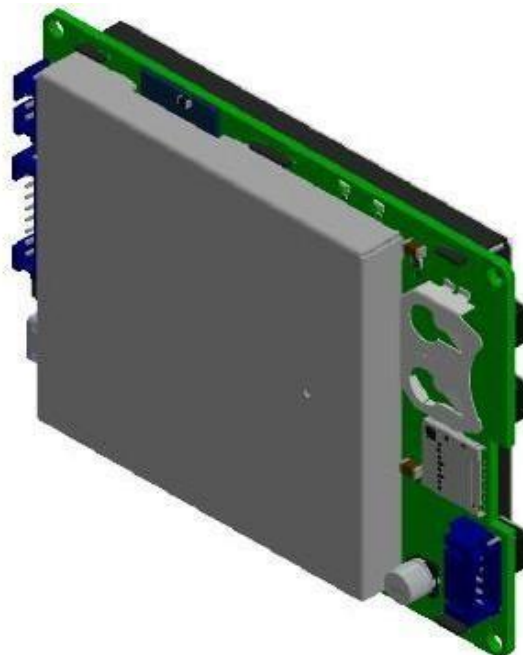
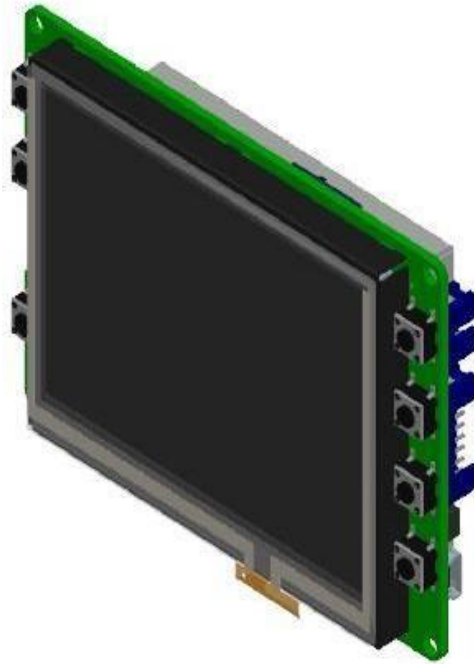
#### 3.1 AGM-035A0-I0 Rev 1.1 (Tela 3,5")



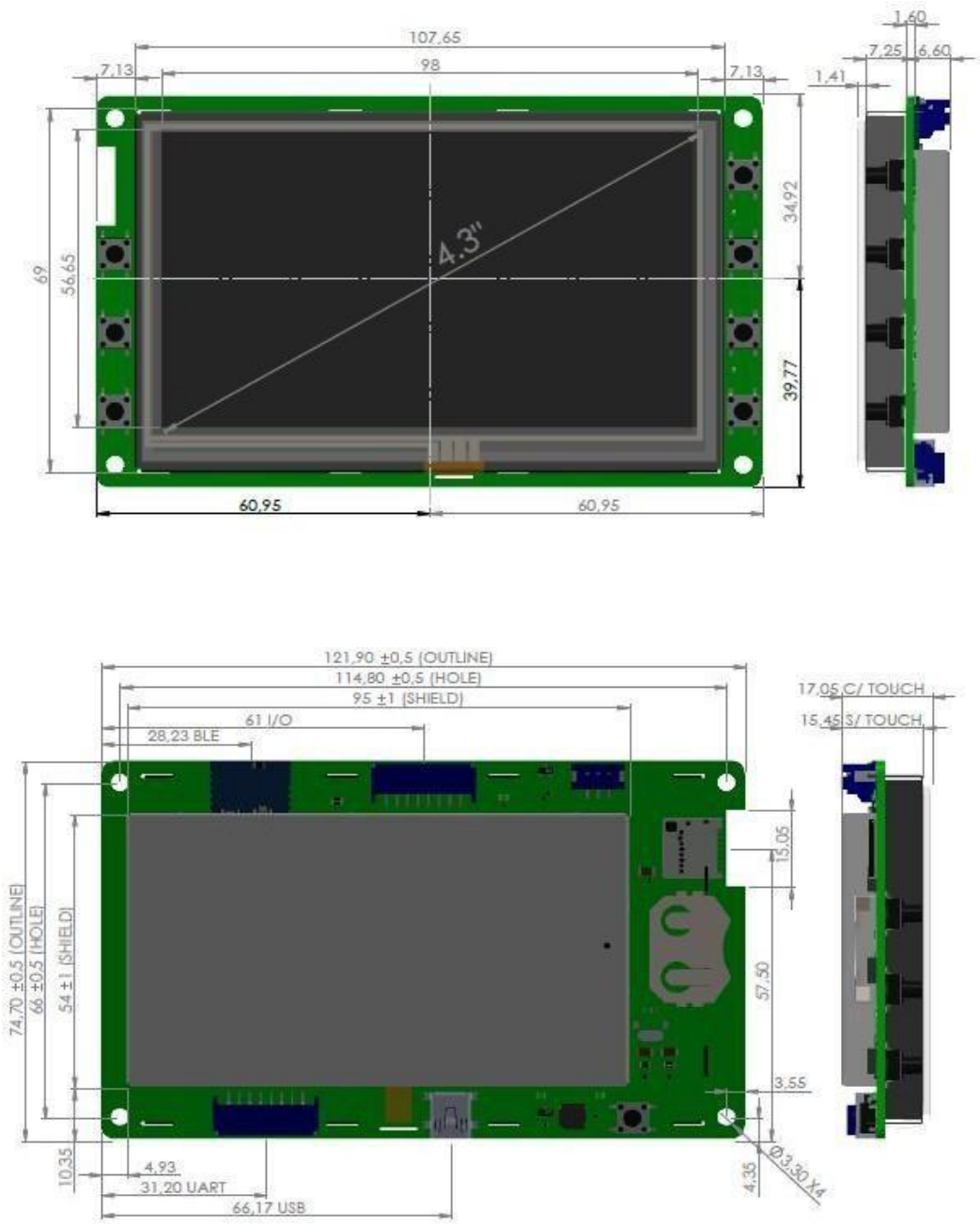
### 3.1 AGM-035A0-I0 Rev 1.1 (Tela 3,5")



### 3.1 AGM-035A0-I0 Rev 1.1 (Tela 3,5")

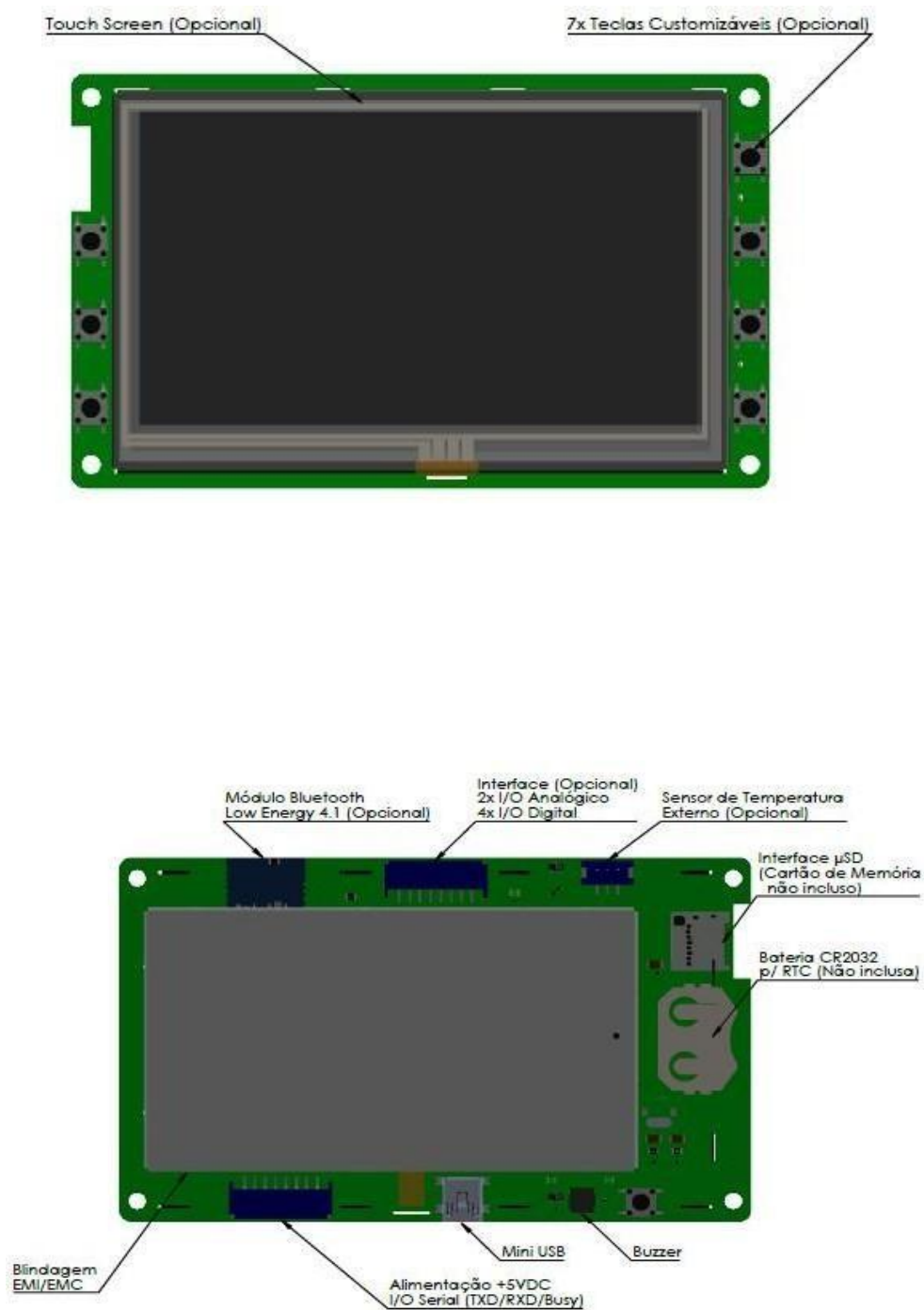


3.2 AGM-043A0-I0 Rev 1.1 (Tela 4,3")

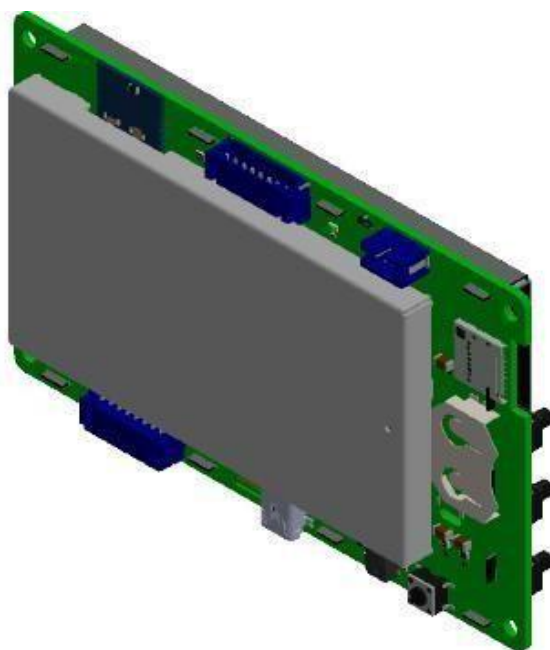
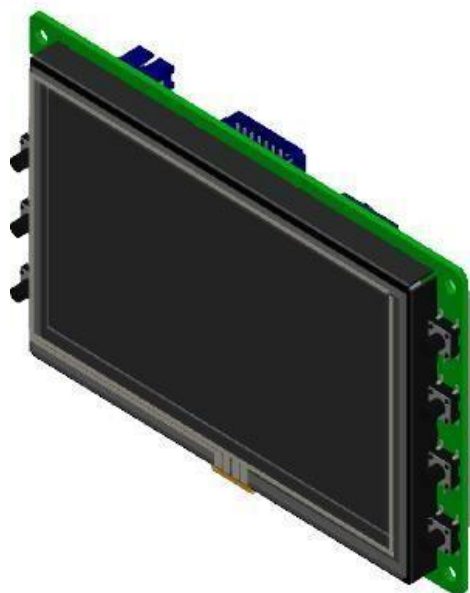




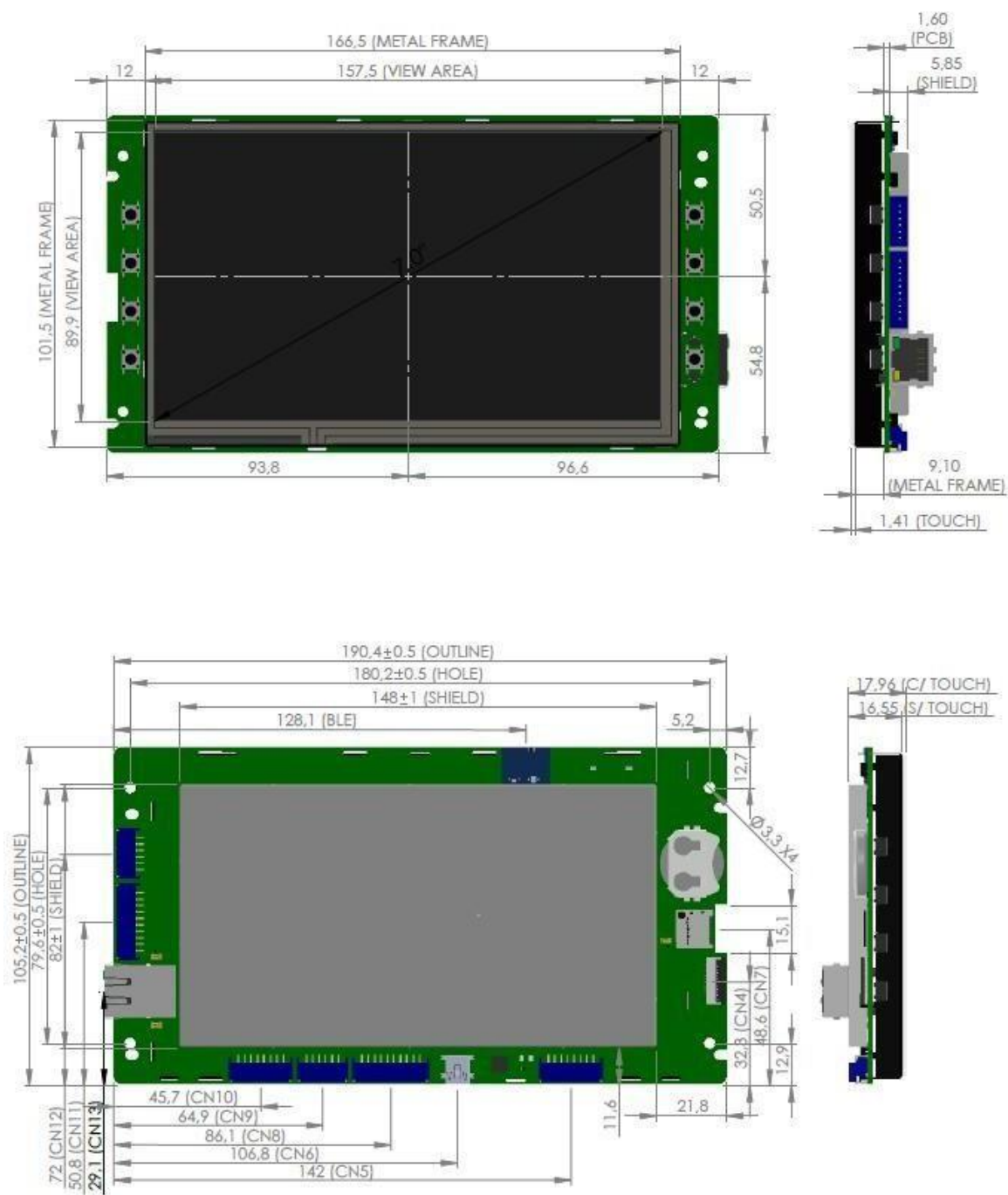
### 3.2 AGM-043A0-I0 Rev 1.1 (Tela 4,3")



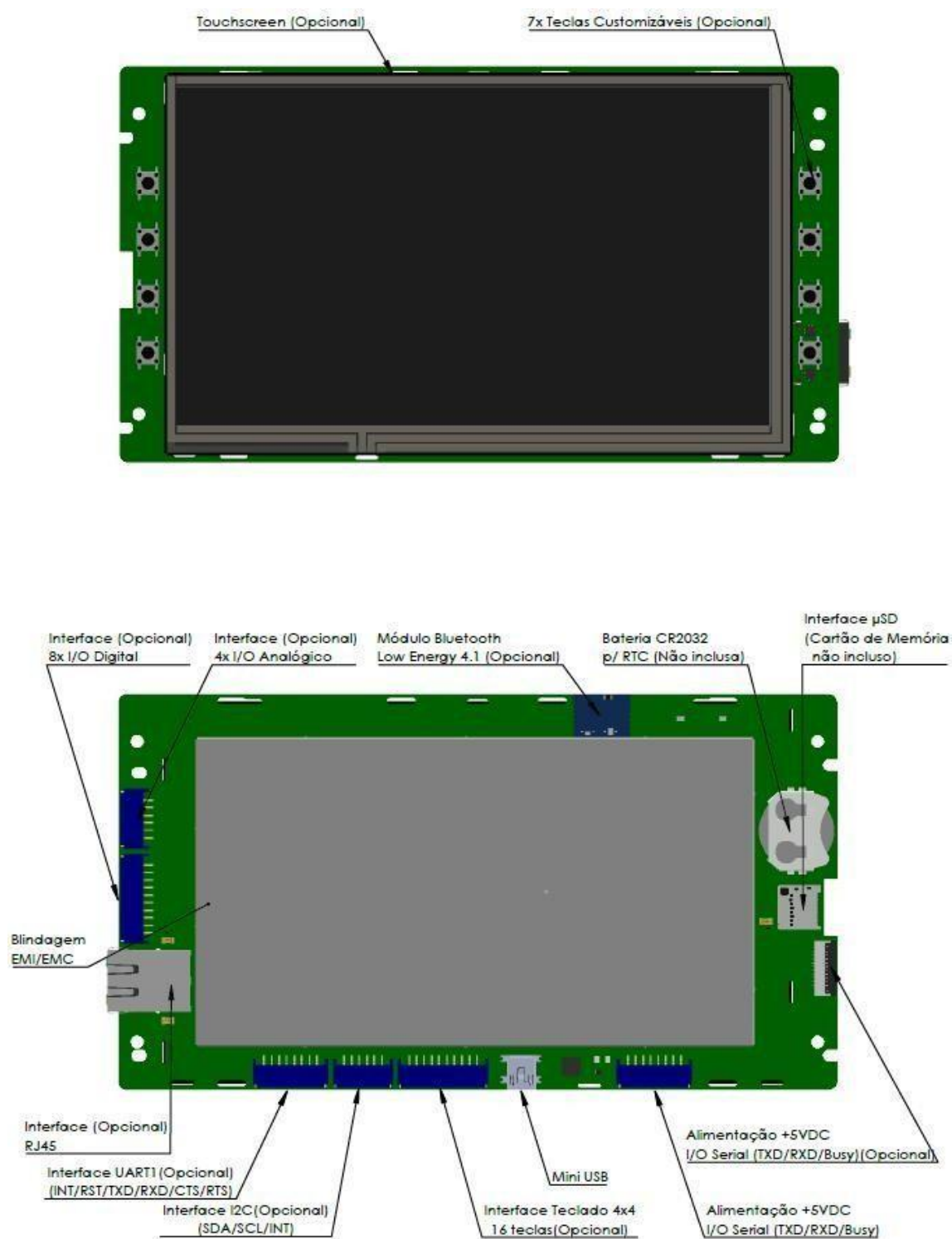
### 3.2 AGM-043A0-I0 Rev 1.1 (Tela 4,3")



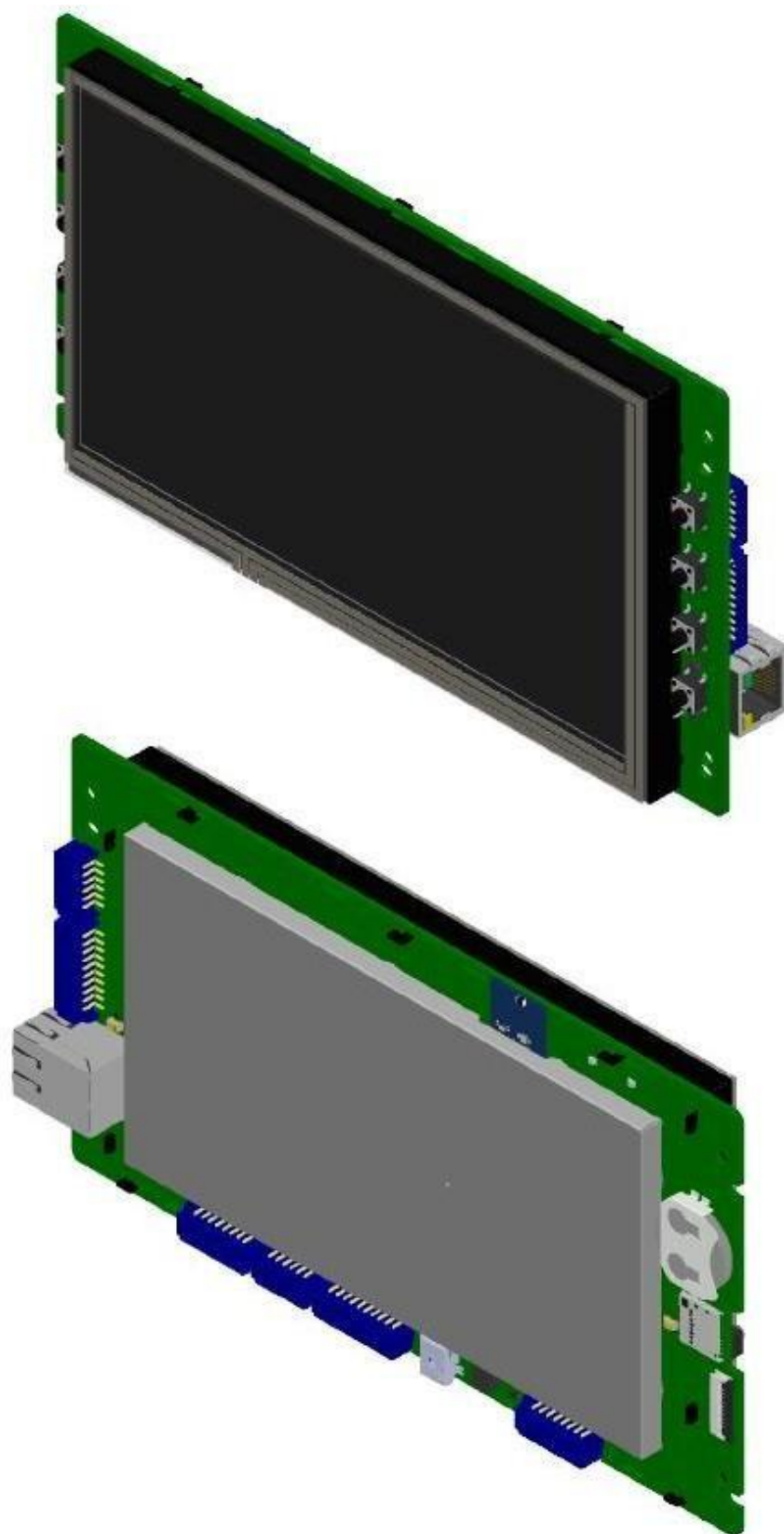
3.3 AGM-070A0-I0 Rev 1.1 (Tela 7,0")



### 3.3 AGM-070A0-I0 Rev 1.1 (Tela 7,0")



### 3.3 AGM-070A0-I0 Rev 1.1 (Tela 7,0")



## 4 Protocolo de Comunicação

O protocolo de comunicação opera, na versão atual do documento, como sendo o descrito a seguir, no modo comando-resposta (comando = enviado pelo equipamento, resposta = gerada pelo IHM):

<HEADER> <LEN> <CMD> <DATA>

Campo	Valores	Descrição
<HEADER>	<0x5A> <0xA5>	Marcadores de início de dados
<LEN>	<0x01> a <0x7F>	Tamanho do campo CMD+DATA
<CMD>	<0x80> <0x02>	Comando para controle do beep interno
	<0x80><0x03>	Comando para pular de tela
	<0x80><0x08>	Comando de habilitação do touch
	<0x81><0x05>	Comando para leitura de status do touch
	<0x81><0x07>	Comando para leitura da posição atual do touch
	<0x82>	Comando de escrita VP (numérico)
	<0x83>	Comando de leitura de VP (numérico)
	<0x84>	Comando para desenhar curva
	<0x85>	Comando de calibração do display
	<0x86>	Comando para iniciar uma animação de tela
	<0x87>	Comando para iniciar uma animação de imagem
	<0x88>	Comando para desenhar linha na tela
	<0x89>	Comando para desenhar círculo na tela
	<0x90>	Comando para desenhar um ponto na tela
	<0x91>	Comando para limpar um pedaço da tela
	<0x92>	Comando para desenhar um polígono na tela
	<0x93>	Comando para desenhar retângulo na tela
<DATA>	-x-	Valores de parâmetros de cada comando, podendo variar de 1 a 128 bytes

O protocolo opera no modo UART 8N1 a 9600 bps. Sendo a taxa de comunicação configurável.

Os comandos <0x82> e <0x83> – são comandos de dados - são aqueles nos quais são possíveis fazer o envio e leitura de dados da IHM, que permitem modificar um número ou alguma informação de um componente existente na tela. Para identificar o campo, usa-se o parâmetro VP dos componentes.

Existe uma opção na qual uma resposta é enviada sem ser solicitada – o que caracteriza um evento. Essa operação ocorre para o comando <0x81><0x07>, no qual o dado das coordenadas de um acionamento de touch é enviado automaticamente sem ser solicitado.

## 4.1 Comandos de Operação

### 4.1.1 Comando para controle do beep interno

Esse comando ativa o beep por um intervalo de tempo configurável.

Comando: <0x5A><0xA5><0x03><0x80><0x02><t>

<t> = tempo de acionamento, em intervalo de 200ms

### 4.1.2 Comando para pular uma tela

Esse comando solicita o salto para uma determinada tela.

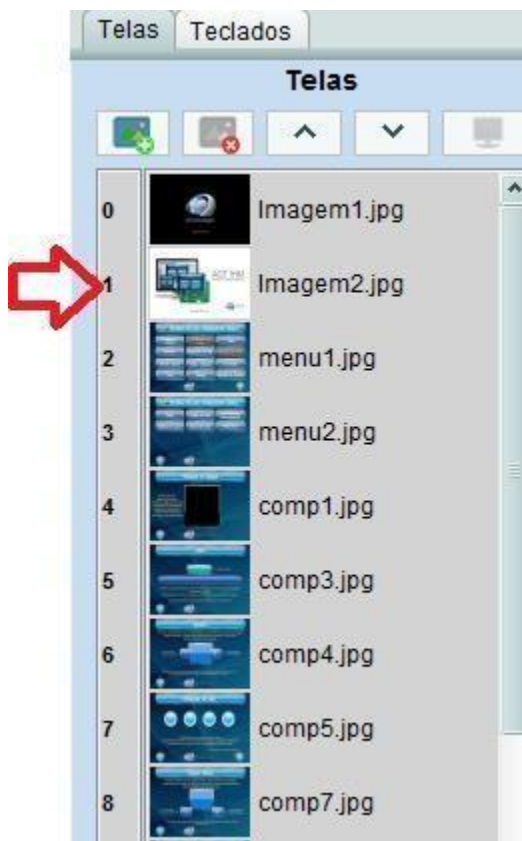
Comando: <0x5A><0xA5><0x04><0x80><0x03><IDH><IDL>

<IDH> = número de identificação da tela (ID), 8 bits mais significativos

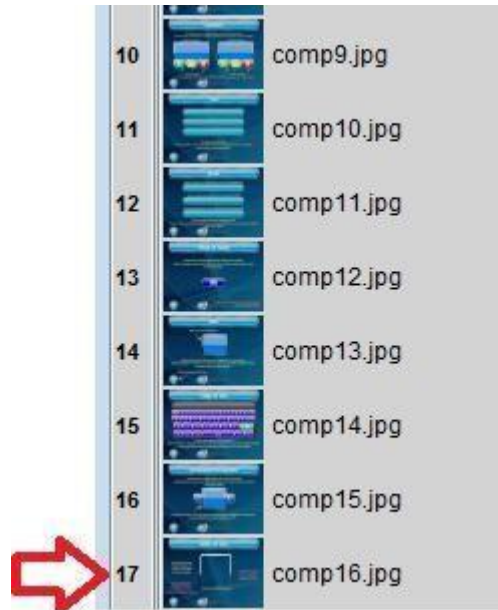
<IDL> = número de identificação da tela (ID), 8 bits menos significativos

O IDH e IDL devem ser em hexadecimal, então, é necessário fazer a conversão dos números das telas decimais para hexadecimais.

Por exemplo:



Caso queira pular para a segunda tela, como mostra a imagem acima, o IDH deve ser <0x00> e IDL <0x01>. Ou então, caso queira mudar para a décima sétima tela, como mostra a imagem a seguir, IDH deve ser <0x00> e IHL <0x11>.



#### 4.1.3 Comando de habilitação do touch

Esse comando habilita ou desabilita o touch screen.

Comando: <0x5A><0xA5><0x03><0x80><0x08><h>

<h> = desabilita o touch se for 0, habilita se for qualquer outro valor

#### 4.1.4 Comando para leitura de status do touch

Esse comando retorna se houve mudança na posição do touch desde a última leitura.

Comando: <0x5A><0xA5><0x02><0x81><0x05>

Resposta do IHM: <0x5A><0xA5><0x03><0x81><0x05><s>

<s> = <0x5A> se houve mudança e <0x00> em caso contrário

#### 4.1.5 Comando para leitura da posição atual do touch

Esse comando retorna a posição atual na qual houve um touch.

Comando: <0x5A><0xA5><0x02><0x81><0x07>

Resposta do IHM: <0x5A><0xA5><0x06><0x81><0x07><XH><XL><YH><YL>

<XH> = 8 bits mais significativos da coordenada X

<XL> = 8 bits menos significativos da coordenada X

<YH> = 8 bits mais significativos da coordenada Y

<YL> = 8 bits menos significativos da coordenada Y



#### 4.1.6 Comando para sinalizar eventos

Esse comando retorna o código configurado no componente. O evento pode ser do tipo Padrão (X,Y) ou Código de Tecla.

Para retornar o evento com código de tecla, o protocolo recebido será:  
<0x5A><0xA5><0x02><0x81><0xCódigo de Tecla configurado>.

Para retornar o evento no modo Padrão (X,Y) o protocolo recebido será:  
<0x5A><0xA5><0x06><0x81><XH><XL><YH><YL>, o qual:

<XH> = 8 bits mais significativos da coordenada X

<XL> = 8 bits menos significativos da coordenada X

<YH> = 8 bits mais significativos da coordenada Y

<YL> = 8 bits menos significativos da coordenada Y

## 4.2 Comando de Dados

### 4.2.1 Comando de escrita de VP (numérico)

Esse comando grava um dado em uma posição de memória interna (VP). Esse dado é somente numérico, limitado a 2 bytes.

Comando: <0x5A><0xA5><LEN><0x82><VPH><VPL><Valor>

<LEN> = 8 bits que indicam a quantidade de bytes que formam o comando.

<VPH> = 8 bits mais significativos do endereço do VP.

<VPL> = 8 bits menos significativos do endereço do VP.

<Valor> = Se for enviar uma palavra de 16 bits vai mandar o <ValH><ValL>

<ValH>= 8 bits mais significativos do valor.

<ValL>=8 bits menos significativos do valor.

Se for de 32 bits vai mandar <ValH><Val\_IH><Val\_IL><ValL>

<ValH> = 8 bits mais significativos do valor.

<Val\_IH> = 8 bits intermediários mais significativos do valor.

<Val\_IL> = 8 bits intermediários menos significativos do valor.

<ValL> = 8 bits menos significativos do valor.

### 4.2.2 Comando de leitura de VP (numérico)

Esse comando lê um dado em uma posição de memória interna (VP). Esse dado é somente numérico, limitado a 2 bytes.

Comando: <0x5A><0xA5><0x04><0x83><VPH><VPL><NB>

<VPH> = 8 bits mais significativos do endereço do VP

<VPL> = 8 bits menos significativos do endereço do VP

<NB> = Número de bytes a ser lido

Resposta: <0x5A><0xA5><0x04+NB><0x83><VPH><VPL><NB><Valor>

<Valor> = Se for enviar uma palavra de 16 bits vai mandar o <ValH><ValL>

<ValH>= 8 bits mais significativos do valor.

<ValL>=8 bits menos significativos do valor.

<Valor> = Se for de 32 bits vai mandar <ValH><Val\_IH><Val\_IL><ValL>  
 <ValH> = 8 bits mais significativos do valor.  
 <Val\_IH> = 8 bits intermediários mais significativos do valor.  
 <Val\_IL> = 8 bits intermediários menos significativos do valor.  
 <ValL> = 8 bits menos significativos do valor.

#### 4.2.3 Comando curva

Esse comando permite o desenho de curvas.

Comando: <0x5A><0xA5><0x04><0x84><C><Vh><Vl>

<C> = comando canal da curva.

<Vh> = 8 bits mais significativos.

<Vl> = 8 bits menos significativos.

#### 4.2.4 Comando de calibração

Esse comando permite a calibração do display.

Comando: <0x5A><0xA5><0x01><0x85>

#### 4.2.5 Comando de iniciar animação de tela

Esse comando permite a iniciação de uma animação na tela.

Comando: <0x5A><0xA5><0x01><0x86>

#### 4.2.6 Comando de iniciar animação de imagem

Esse comando permite a iniciação de uma animação de imagem.

Comando: <0x5A><0xA5><0x01><0x87>

#### 4.2.7 Comando para desenhar linha na tela

Esse comando permite o desenho de uma linha na tela.

Comando:

<0x5A><0xA5><0x0C><0x88><HColor><MColor><LColor><X0><X0\_1><Y0><Y0\_1><XF><XF\_1>  
 <YF><YF\_1>

<HColor> = 8 bits que indicam o byte mais significativo da cor.

<Mcolor> = 8 bits que indicam o byte intermediário da cor .

<LColor> = 8 bits que indicam o byte menos significativo da cor.

<X0> = O byte mais significativo da posição inicial de x da linha.

<X0\_1>= O byte menos significativo da posição inicial de x da linha.

<Y0> = O byte mais significativo da posição inicial de y da linha.

<Y0\_1> = O byte menos significativo da posição inicial de y da linha.

<XF> = O byte mais significativo da posição final de x da linha.

<XF\_1> = O byte menos significativo da posição final de x da linha.

<YF> = O byte mais significativo da posição final de y da linha.

<YF\_1> = O byte menos significativo da posição final de y da linha.

#### 4.2.8 Comando para desenhar círculo na tela

Esse comando permite o desenho de um círculo na tela.

Comando:

<0x5A><0xA5><0x0B><0x89><HColor><MColor><LColor><Fill><XCentro><XCentro\_1><YCentr  
 o><YCentro\_1><Raio><Raio\_1>

<Hcolor> = 8 bits que indicam o byte mais significativo da cor.

<Mcolor> = 8 bits que indicam o byte intermediário da cor.

<L color> = 8 bits que indicam o byte menos significativo da cor.

<Fill> = Mandar <0x00> ou <0x01>. 0=não preenchido e 1=preenchido.

<XCentro> = O byte mais significativo da posição central do x.  
 <XCentro\_1> = O byte menos significativo da posição central do x.  
 <YCentro> = O byte mais significativo da posição central do y.  
 <YCentro\_1> = O byte menos significativo da posição central do y.  
 <Raio> = O byte mais significativo do raio da circunferência.  
 <Raio\_1> = O byte menos significativo do raio da circunferência.

#### 4.2.9 Comando para desenhar um ponto na tela

Esse comando permite o desenho de um ponto na tela.

Comando: <0x5A><0xA5><0x08><0x90><HColor><MColor><LColor><X><X1><Y><Y1>

<Hcolor> = 8 bits que indicam o byte mais significativo da cor.

<Mcolor> = 8 bits que indicam o byte intermediário da cor.

<L color> = 8 bits que indicam o byte menos significativo da cor.

<X> = O byte mais significativo da posição do x.

<X1> = O byte menos significativo da posição x.

<Y> = O byte mais significativo da posição y.

<Y1> = O byte menos significativo da posição y.

#### 4.2.10 Comando para limpar tela

Esse comando para limpar um pedaço da tela.

Comando: <0x5A><0xA5><0x09><0x91><X0><X0\_1><Y0><Y0\_1><XF><XF\_1><YF><YF\_1>

<X0> = O byte mais significativo da posição inicial de x da tela.

<X0\_1> = O byte menos significativo da posição inicial de x da tela.

<Y0> = O byte mais significativo da posição inicial de y da tela.

<Y0\_1> = O byte menos significativo da posição inicial de y da tela.

<XF> = O byte mais significativo da posição final de x da tela.

<XF\_1> = O byte menos significativo da posição final de x da tela.

<YF> = O byte mais significativo da posição final de y da tela.

<YF\_1> = O byte menos significativo da posição final de y da tela.

#### 4.2.11 Comando para desenhar um polígono

Esse comando permite o desenho de um polígono.

Comando: <0x5A><0xA5><LEN><0x92><N><HColor><MColor><LColor><Fill><X><X1><Y><Y1>

<LEN> = 8 bits que indicam a quantidade de bytes que formam o comando

<N> = Os pares das coordenadas dos pontos do polígono. Ex: Enviar três pontos.

<XH><XL><YH><YL><X1H><X1L><Y1H><Y1L><X2H><X2L><Y2H><Y2L>.

<Hcolor> = 8 bits que indicam o byte mais significativo da cor.

<Mcolor> = 8 bits que indicam o byte intermediário da cor.

<L color> = 8 bits que indicam o byte menos significativo da cor.

<Fill> = Mandar 0 ou 1. 0=não preenchido e 1=preenchido.

<X> = O byte mais significativo do X da origem do polígono.

<X1> = O byte menos significativo do X da origem do polígono.

<Y> = O byte mais significativo do Y da origem do polígono.

<Y1> = O byte menos significativo do Y da origem do polígono.

#### 4.2.12 Comando para desenhar um retângulo

Esse comando permite desenhar um retângulo.

Comando: <0x5A><0xA5><0x0D><0x93><HColor><MColor><LColor><Fill><X0><X0\_1><Y0><Y0\_1><XF><XF\_1><YF><YF\_1>

<Hcolor> = 8 bits que indicam o byte mais significativo da cor.  
 <Mcolor> = 8 bits que indicam o byte intermediário da cor.  
 <L color> = 8 bits que indicam o byte menos significativo da cor.  
 <Fill> = Mandar <0x00> ou <0x01>. 0=não preenchido e 1=preenchido.  
 <X0> = O byte mais significativo da posição inicial de x do retângulo.  
 <X0\_1> = O byte menos significativo da posição inicial de do retângulo.  
 <Y0> = O byte mais significativo da posição inicial de y do retângulo.  
 <Y0\_1> = O byte menos significativo da posição inicial do retângulo.  
 <XF> = O byte mais significativo da posição final de x do retângulo.  
 <XF\_1> = O byte menos significativo da posição final de do retângulo.  
 <YF> = O byte mais significativo da posição final de y do retângulo.  
 <YF\_1> = O byte menos significativo da posição final do retângulo.

#### 4.2.14 Comando para atualizar o relógio

Esse comando permite atualizar hora e data do relógio.

Comando: <0x5A><0xA5><0x0B><0x80><0x1F><0x5A><AnoH><AnoL><Mês><Dia\_do\_mês><Dia\_da\_Semana><Hora><Minuto><Segundo>.

<Ano> = A configuração do ano vai ser feita a partir dos parâmetros <AnoH> e <AnoL>, sendo o 8 bits mais significativos e 8 bits menos significativos, respectivamente.

<Mês> = O mês deve ser inserido em hexadecimal.

<Dia\_do\_mês> = O dia do mês deve ser inserido em hexadecimal.

<Dia\_da\_semana> = Para configurar o dia da semana, deve ser colocado o comando de acordo com a necessidade:

<0x00> = Domingo  
 <0x01> = Segunda-feira  
 <0x02> = Terça-feira  
 <0x03> = Quarta-feira  
 <0x04> = Quinta-feira  
 <0x05> = Sexta-feira  
 <0x06> = Sábado

<Hora> = Para atualizar a hora, nesse protocolo, deve inserir as horas entre 0 horas e 23 horas desejada em hexadecimal.

<Minuto> = Para atualizar os minutos, nesse protocolo, deve inserir os minutos entre 0 minutos e 59 minutos desejado em hexadecimal.

<Segundo> = Para atualizar os segundos, nesse protocolo, deve inserir os segundos desejados entre 0 segundos e 59 segundos em hexadecimal.

#### 4.2.15 Comando para leitura relógio

Esse comando permite a leitura da hora e data do relógio.

Comando: <0x5A><0xA5><0x03><0x81><0x20><0x26>

Resposta: <0x5A><0xA5><0x0B><0x81><0x20><0x26><AnoH><AnoL><Mês><Dia\_do\_mês><Dia\_da\_semana><Hora><Minuto><Segundo>.

<Ano> = A configuração do ano vai ser feita a partir dos parâmetros <AnoH> e <AnoL>, sendo o 8 bits mais significativos e 8 bits menos significativos, respectivamente.

<Mês> = O mês deve ser inserido em hexadecimal.

<Dia\_do\_mês> = O dia do mês deve ser inserido em hexadecimal.

<Dia\_da\_semana> = Para configurar o dia da semana, deve ser colocado o comando de acordo com a necessidade:

<0x00> = Domingo  
<0x01> = Segunda-feira  
<0x02> = Terça-feira  
<0x03> = Quarta-feira  
<0x04> = Quinta-feira  
<0x05> = Sexta-feira  
<0x06> = Sábado

<Hora> = Para atualizar a hora, nesse protocolo, deve inserir as horas entre 0 horas e 23 horas desejada em hexadecimal.

<Minuto> = Para atualizar os minutos, nesse protocolo, deve inserir os minutos entre 0 minutos e 59 minutos desejado em hexadecimal.

<Segundo> = Para atualizar os segundos, nesse protocolo, deve inserir os segundos desejados entre 0 segundos e 59 segundos em hexadecimal.

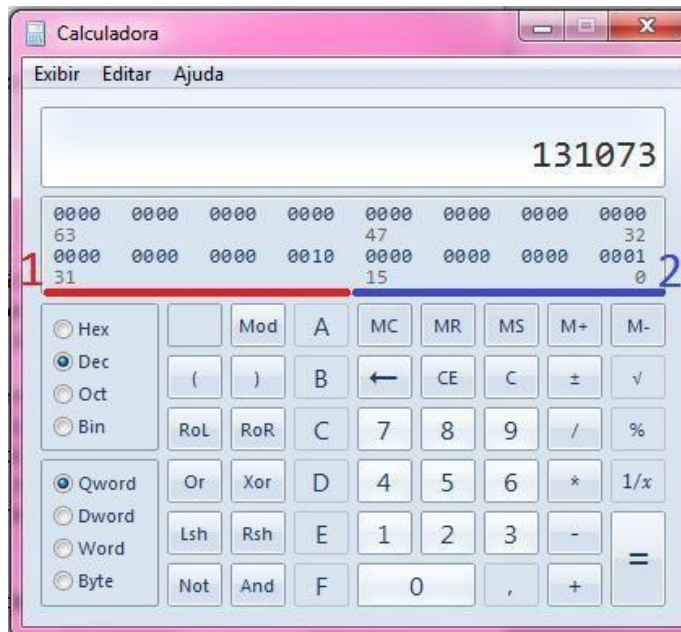
#### **4.2.16 Modo de utilização do VP**

Cada VP utilizado pela da IHM guarda apenas informações de 16 bits. Portanto, quando utilizado o componente que têm a opção onde pode ser configurado se será usado o número inteiro (16 bits), inteiro (32 bits) ou não inteiro (32 bits), a IHM trabalha da seguinte forma:

Quando usado a opção de inteiro (16 bits), o valor é salvo somente no VP configurado. Quando usado inteiro (32 bits) ou não inteiro (32 bits), o valor é salvo pela IHM em dois endereços, pois como o VP armazena informações de 16 bits são utilizados dois endereços: o VP configurado e o subsequente a ele. A parte mais significativa do número no VP configurado e a parte menos significativa no subsequente. Para que faça a leitura do valor utilizado no componente, deve-se ler o VP configurado e o próximo.

Exemplificando:

Caso queira que o VP salve o valor de 131.073 no VP [0x01], deve ser utilizado 32 bits:



Este valor é “divido” a cada 16 bits, sendo então:

- A parte vermelha, marcado como 1, são os bits mais significativos. Na conversão de binário para decimal, obtem-se o valor 2, o qual será encontrado no VP desejado, ou seja, VP[0x01];
- A parte roxa, marcado como 2, são os bits menos significativos. Na conversão de binário para decimal, obtem-se o valor 1, o qual será encontrado no VP seguinte do VP dos bits mais significativos, ou seja, VP[0x02];