Touch and graphical displays part 3 - Graphics and widgets

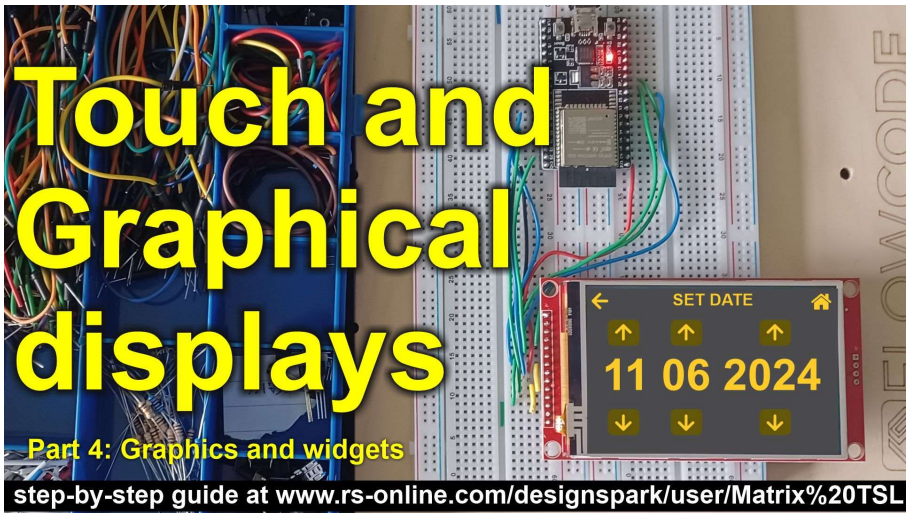

Figure 1 - Youtube video link

The project

In this article we are going to look at how to design graphics and touch screen widgets for a touch screen menu system using Flowcode. We will look at:

- The design process
- Colours you can use
- Bitmap graphical elements
- Text and fonts
- Vector graphical elements
- Touch screen widgets

Let's start with the design process:

Design process

We have to do several things here:

- Decide on the design style
- Understand the technical design elements
- Plan the project

A key part of this is that in many industrial projects there will be a number of stakeholders that want to get involved in the design of the touch screen system: the engineering team will be very technical and costs conscious, the marketing and sales department will have opinions about styles, logos, and features, all users will be conscious of ergonomics and ease of use. For this reason a key element of the project has to be a way of communicating what the options are for the team and what the project will look like before any programming work starts.

Take away 1: planning your project and developing a system of communicating your intentions to others is an important task ahead of you.

Even if you are just a team of 1, planning your project before you start programming will save time in the long run.

Design styles

The first thing we need to do is establish what kind of style of graphics we will use for our project. There are trade-offs and options here. For the sake of discussion we have created a project that will demonstrate the decision making and design process: a weather and heating control / monitoring station with touch screen. If your design team includes a graphic designer and an embedded engineer, it might be good to present initial design ideas to the rest of the team. They can be produced in Indesign, Corel or any other package. If you are a team of 1 going through a process of deciding your design style will also be necessary.

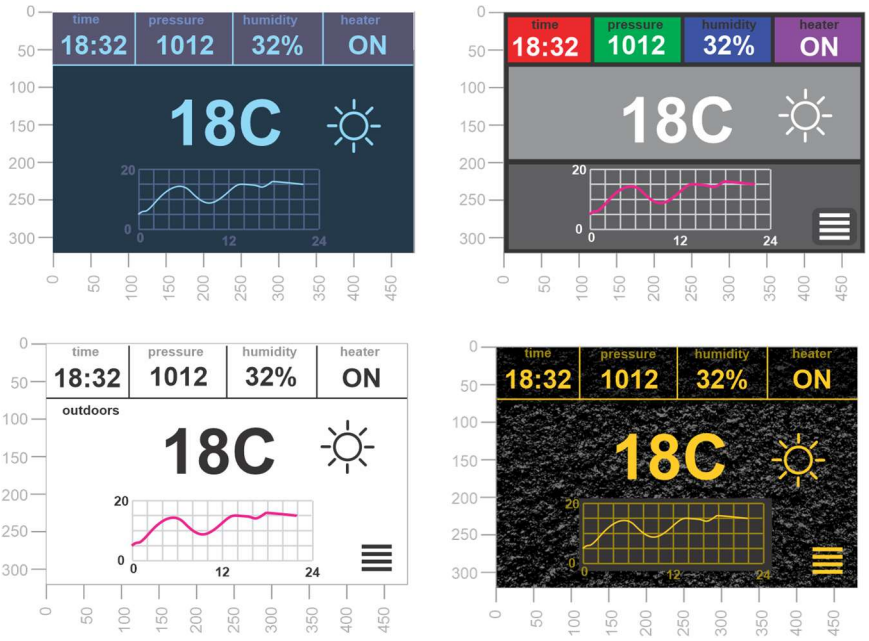

Figure 2 - initial design options

In our project the team has chosen a 480 x 320 capacitive touch screen display. This choice is governed by the size of the housing, cost, ergonomics and perhaps other factors.

The embedded engineer and the designer would come up with some initial ideas for the rest of the team to sanction. These initial designs illustrate the main display of the weather monitor project: a main temperature display, a graph of daily temperature, a graphic for the current weather status - sunny in this case - and four top panels for time, pressure, humidity and heater status. It helps design process and discussion to have the x-y pixel dimensions marked on the outside of the display. We have shown here 4 different designs with the same information:

- Top left: 'Cool blue': relatively simple style based on multiple shades of blue for different parts of the display.
- Top right: 'Windows x' More colourful version, but the same graphical elements a bit like how Windows used to present their main screen.
- Bottom left: 'Paper white': simple minimal colour display.
- Bottom right: 'Marble': a design that is lifted by the inclusion of a background stone-like bitmap.

As you can see the same information can be presented in different graphical styles. Much of this comes down to personal preference but there are some technical issues:

The temptation for all will be to go down a route that is as graphically attractive as possible. But this is a big mine field.

Take away 2: you can spend a really big chunk of your project budget just messing around with graphics for touch screens. It is great fun. But it is most likely not in the budget, and its rarely the main project concern. Keeping it simple is the key.

Our first three mockups are similarly constructed. A style of design like the 'Marble' design at the bottom right presents some challenges. Let's take a closer look:

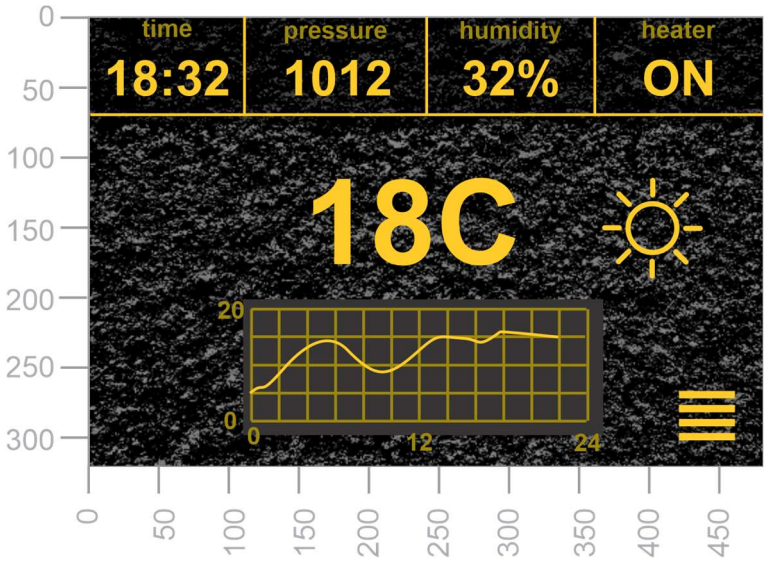

Figure 3 - Marble home screen design

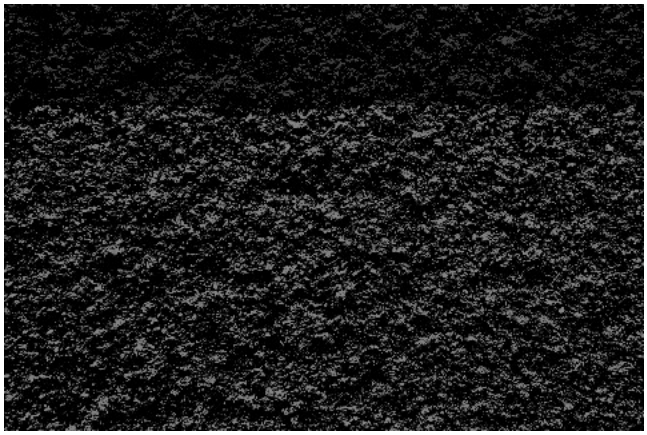

Figure 4 - Marble background bitmap

The Marble design is made up of several elements. The first element is a background bitmap. This is shaded in two parts to differentiate the different areas of the display. One of the drawbacks here is that a full 24 bit colour bitmap like this takes some time to display. Its not so important in our application as weather data is changing slowly, but if you had an application where the data was moving quickly this might be an issue as whenever a number changed the whole screen would have to be redrawn.

The technology behind your display system matters here: if you are on a slow SPI connection it might take a few seconds to load this bitmap. If you are on a fast a parallel display it might take less than a second. You can also get round this by drawing a single colour rectangle ‘under’ the space where the text data will change often - in this case a black rectangle would look fine. We have done this with the graph area in figure 3

The next elements are the graphics used in the touch graphics. In this case the sun and menu icons. There are two possible techniques here. Firstly you can add the Marble texture to the graphics to make them appear to float on top of the background. We have done that here:

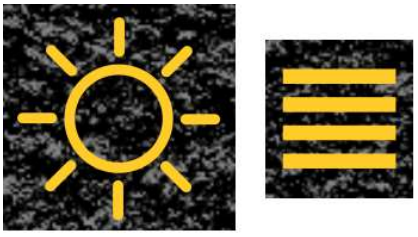

Figure 5 - Sun and touch icons

It's also possible to specify a colour in the bitmap to use as a transparent mask, for example black or white could be used as transparency. This isn't standard but is an option in Flowcode.

The Marble design here is a relatively simple example of a display that uses bitmaps to add texture and next level design. Other designs use graphics to present the user with a game-like 3D render of a physical controller. These are possible, but they add a level of complexity, and time, to your project and require tighter co-ordination between the graphics team and the programming team so go down this route with caution.

All the other elements of the display are vector based: lines, text, graph.

Take away 3: reducing the number of bitmaps and keeping your design to use blocks of colour will save you time.

Technical side of design elements

The next step is that the team need to understand the technical limitations.

The Flowcode Graphical Display Manager allows you to create three different types of elements that can be controlled by the component macros:

1. Touch Widgets
2. Text fields
3. Objects

Touch widgets

These are as follows:

Widget_Rectangle - 1 - A rectangular box with a shadow. A simple touchable object. You can set the foreground colour, the outline colour and the shadow colour.

Widget_RoundedRectangle - 2 - A rounded rectangular box with a shadow. A simple touchable object. You can set the foreground colour, the outline colour and the shadow colour.

Widget_Ellipse - 3 - A simple ellipse or circle with a shadow. A simple touchable object. You can set the foreground colour, the outline colour and the shadow colour.

Widget_VSlider - 4 - A vertical slider with a square thumb. You can set the button colour, the background colour and the slider colour.

Widget_HSlider - 5 - A horizontal slider with a square thumb. You can set the button colour, the background colour and the slider colour.

Widget_VSlider_Round - 6 - A vertical slider with a round thumb. You can set the button colour, the background colour and the slider colour.

Widget_HSlider_Round - 7 - A horizontal slider with a round thumb. You can set the button colour, the background colour and the slider colour.

Widget_Hidden - 0 - A hidden area with no graphics. This allows you to use a bitmap as a touchable element: you place the bitmap on the screen and declare the touchable area on top of it.

Widgets can also have text labels. This is really useful for having labelled buttons formed by the non-slider widgets: text on top of a simple rectangle. There are only 7 basic widgets, but with these you can create amazing touch screen systems with them.

Figure 6 - Basic touch widgets with Foreground: Red, Background: Green, Highlight Yellow, Lowlight Blue

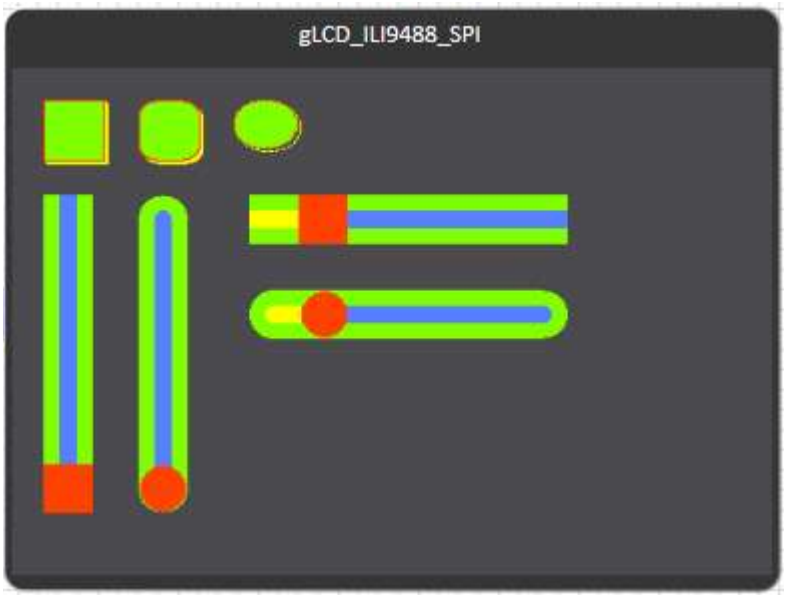

Figure 7 - Navigation and control elements in our chosen style

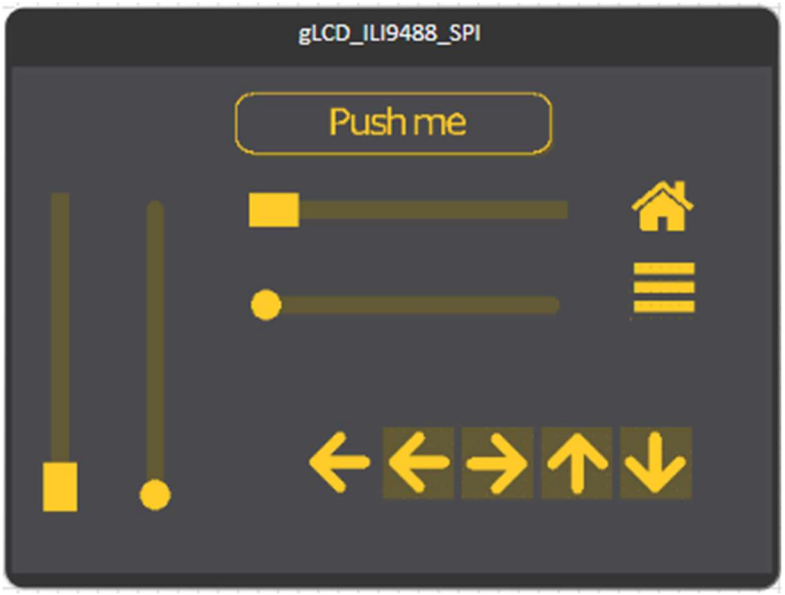

Figure 6 shows widgets in primary colours so you can see the possibilities which parts of the elements can be coloured. Figure 7 shows these design elements in our chosen style - in practice slightly different from the original brief - and some bitmaps used for navigation.

Text fields

Text fields with different font types and sizes are available. The fonts used in text fields and text labels for widgets are controlled by the properties of the Graphical Display component. Fonts are compiled to the microcontroller device during compilation and they are memory hungry. For this reason you can select the number of fonts used and some fonts are numbers only to keep the font size down. Fonts are given a number property which relates to the height in pixels of the characters. So Arial 14 will be 14 pixels high.

Take away 4: In general sans serif fonts (e.g. Arial) display better on a touch screen than serif fonts (e.g. Roman). In general its better to keep your use of fonts to a minimum: lots of different styles in a design looks messy.

Flowcode supports many fonts which are dictated by your selection of fonts in the Display properties. Flowcode focuses on doing a limited number of fonts well rather than providing a comprehensive selection. We like Arial, Arial bold, seven-segment, and Courier bold. Arial and Calibri are nice sans-serif fonts. Seven-segment is great for displays. Courier is a nice proportional font that's good for showing data. Of course there is also a custom font option allowing you to specify any font

you want. If these fonts are not to your liking then there is free font making software on the Flowcode Wiki: this is available but not fully supported as it is freeware and not from the Flowcode team.

The following display shows you the fonts that are in the graphical displays by default:

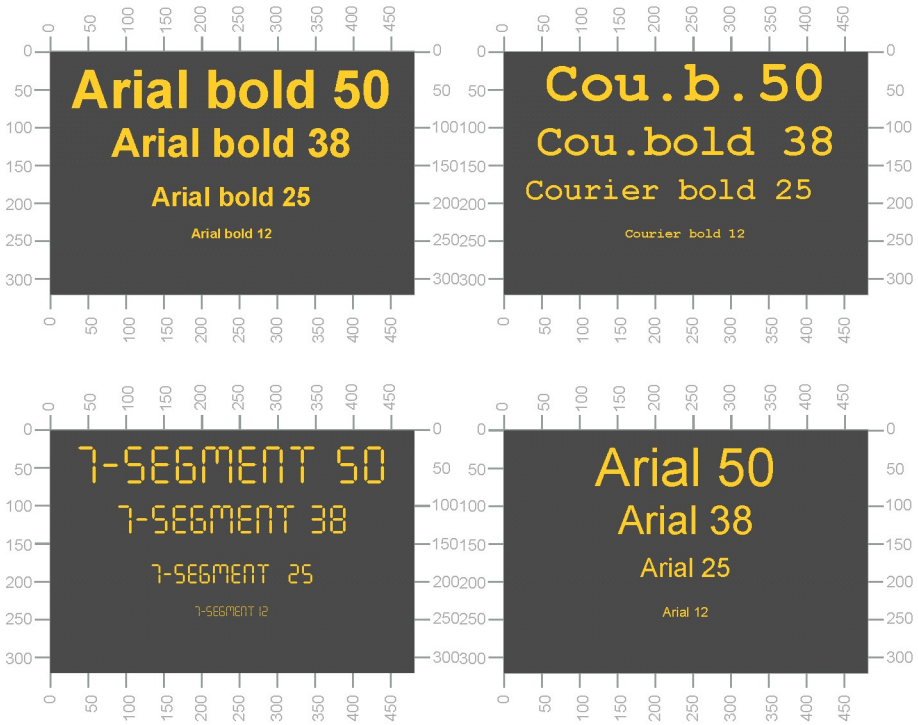

Figure 8 - Default fonts in Flowcode displays.

Font sizes are provides up to size 160 - 160 pixels tall.

Objects

These allow basic graphics to be designed and are as follows:

- **Object_Hidden** - 0 - A hidden area with no graphics.
- **Object_ text** - a text area
- **Object_Rectangle** - 1 - A simple rectangular box.
- **Object_Line** - 2 - A simple straight line.
- **Object_Ellipse** - 3 - A simple ellipse or circle.
- **Object_EllipseFilled** - 4 - A filled ellipse or circle.
- **Object_RectangeFilled** - 5 - A filled rectangular box.

At first glance this looks like it is quite a limited range of graphics objects. But you will be surprised how well they can be used in combination to present good looking touch screen systems.

Planning

Next we need to do a paper based design of the menu system. In practice lets assume that for our design the team discussed graphical styles and they came up with a hybrid style based on the 'Marble 'colour scheme but with a more simple vector based style which would be functionally quicker and technically easier to implement. Having analysed the functionality of the system the graphics designer and embedded programmer come up with a specification as follows:

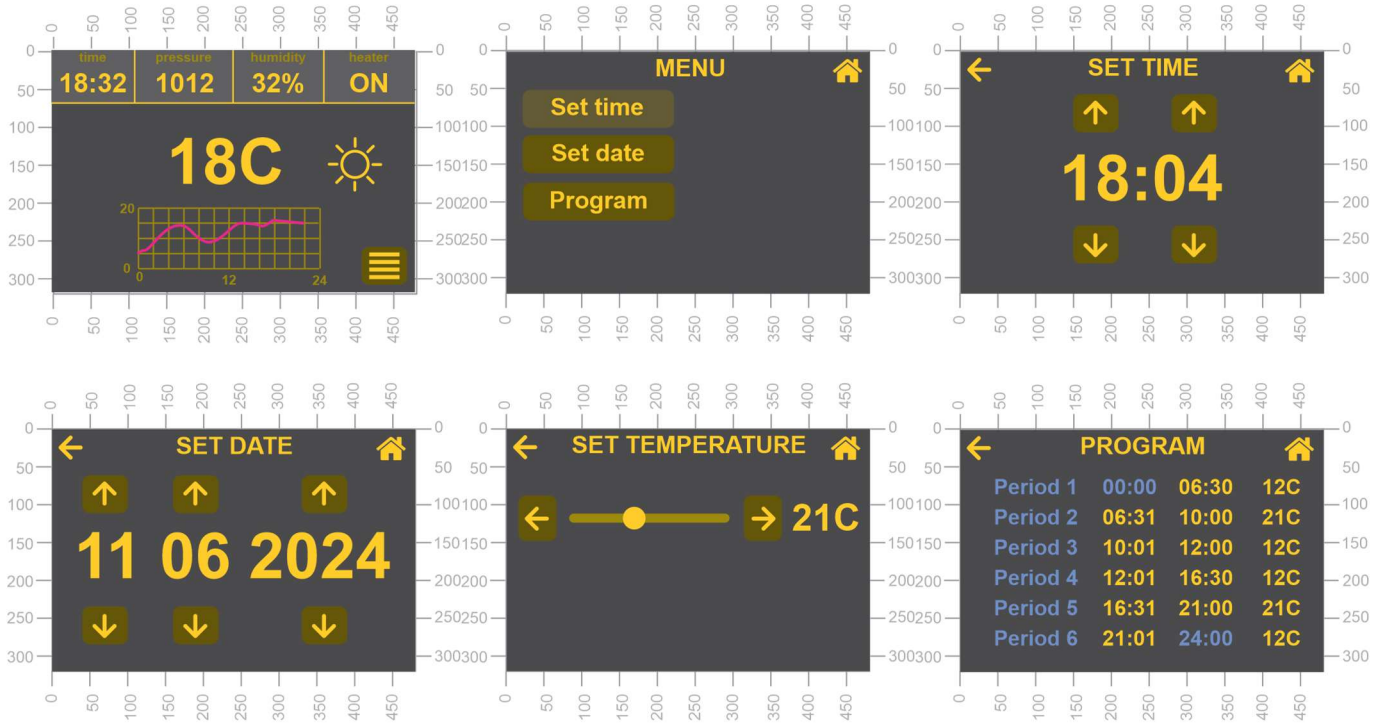

Figure 9 - Functional specification of the system

The great thing about specifying a system in this way is that all stakeholders in the design can see exactly how the menu system is going to work and how it is going to look. From a manager's point of view it also ties the engineers down to a tight specification of work.

Colours and themes

The colour theme will be down to the graphical designer who will understand that in a system like this it is better to use relatively few different colours in one screen. In this scheme the designer has chosen the following colours:

Name	RGB colour	Hex colour BGR
Gold – text	R255 G204 B41	29CCFF
Lowlight gold – dim text, axes	R158 G137 B15	0F899E
Brown - buttons background	R99 G90 B54	365A63
Slate grey - screen background:	R75 G75 B78	4E4B4B
Light grey - top background	R96 G96 B98	626060
Blue - non adjustable text	R113 G143 B200	FF8057

So in this design we are only using 6 colours. They are specified in 24 bit format - 8 bits each for Red Green and Blue. In the table you can see the Decimal RGB colour equivalents and the Hexadecimal equivalents in BGR which is also sometimes used. The colour picker in Flowcode allows you to choose these colours.

Entering these colours every time you create an object is a pain. To make this easier and to facilitate consistency in the project Flowcode groups the colours into themes in the software. There are different themes for Widgets and Objects/Text.

Our themes are:

Widget theme 1 - used for all buttons

Foreground:	Lowlight gold	R158 G137 B15	0F899E
Background:	Brown	R99 G90 B54	365A63
Highlight	Slate grey	R75 G75 B78	4E4B4B

Lowlight	Slate grey	R75 G75 B78	4E4B4B
Text/Object theme 1			
Foreground:	Gold – text	R255 G204 B41	29CCFF
Background:	Brown	R99 G90 B54	365A63
Text/Object theme 2			
Foreground:	Blue	R113 G143 B200	FF8057
Background:	Slate grey	R75 G75 B78	4E4B4B

When you are creating bitmaps to display you will need to use a graphics package. You need to be aware that graphics packages - including Windows Paint - allow you to create bitmaps with different colour depths. Bitmaps can be 1 bit (monochrome), 4 bit, 8 bit and 24 bit. Flowcode supports all of these bitmap variants. For small images it's best to avoid the 4 bit and 8 bit options as the size of the colour pallet can mean the file actually takes longer to load then for a 24 bit image.

Another problem here is that different displays handle colours in different ways. In particular many displays want 24 bit information but only use 5 or 6 bits for each colour. So they will not display colours in the way you planned. You may have to tweak your colours for the display you are using.

Take away 5: Make sure you understand how your graphics package(s) and your hardware handles palettes. You may have to fine tune your colours.

Bitmap and graphical elements

Once the design is agreed the next step is to create the graphical elements. If you search the web you can find hundreds of sheets of icons that are suitable for this so we don't include any in Flowcode. Some icon sheets are free, some are paid for - just a few $. If you don't have a graphic designer you will need some modest graphics skills here. For our design we created the following icons:

Figure 10 - touch control graphics

These are all 45 pixels by 45 pixels. 24 bit BMP. Each one is saved as an individual file and is given a bitmap number.

We also created similar graphics for snow, sun and rain. These are all 70 pixels by 70 pixels. They use colours Gold highlight and Slate grey background. 24 bit BMP. Each one is saved as an individual file.

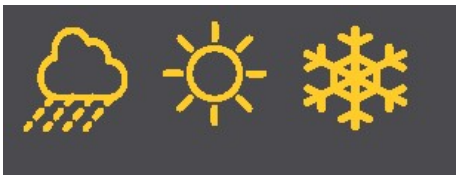

Figure 11 - Weather graphics

Take away 6: Don't use lots of different sizes of graphics - keep it to a minimum and it will save you time.

Remember that keeping your graphics elements to a minimum will save on redraw time for your display.

Once our plan is complete and approved by your team you can start to program.

Touch screen displays for small microcontrollers

The graphics and fonts all take up memory space. That's great if you have something like an ESP32 that has 500k of memory. But if you have a small Arduino processor then it can give you a problem: there just is not the space on the chip. This does not mean that you cant use a small microcontroller: it just means that you need to use different techniques: Rather than including bitmap graphics elements in your design you will need to rely more on 'pressable' button widgets with text and with text characters like: '<', '>','^', etc.

Conclusion

Design of touch screen systems might be a new area for many embedded engineers who may not be used to working with graphics.

For many projects this will be a collaboration between graphics designer, programmer and engineer. Coming up with a plan before programming starts should be a key objective. This will save time in the long run.

The example we have given is quite linear - in practice the first design a team does will be more iterative until all team

members understand the graphical issues, the choice of fonts, the available widgets, objects etc. .

Flowcode is a great software tool for graphical and touch screen display system design, and the simulation capabilities of Flowcode are really useful and save lots of time.

Figure 12 - About the author