Stm32 ile p4 arasında haberleşme protokolleri

STM32 ile P4 LED ekran arasındaki haberleşme protokolleri, doğrudan bir seri veya paralel iletişim protokolü olarak tanımlanmaz; daha ziyade, özel bir arayüz standardı olan HUB75 protokolüne dayanır. STM32 mikrokontrolcüsünün bu paneli sürmek için kullanması gereken yöntemler ve bu haberleşme için kritik olan pinler ve sinyaller aşağıda detaylandırılmıştır.

HUB75 Protokolü ve Sinyalleri

HUB75, LED matris panellerini kontrol etmek için yaygın olarak kullanılan 16 pinli bir arayüzdür. P4 ve P5 panellerin her ikisi de bu standart arayüzü kullanır, bu nedenle fiziksel olarak aynı kablo ve soket yapısını paylaşırlar. Ancak, panelin özelliklerine (piksel boyutu, tarama oranı vb.) göre pinlerin yazılımda nasıl sürüleceği farklılık gösterebilir.

HUB75 arayüzündeki pinler üç ana gruba ayrılır:

1. **Veri Pinleri (R1, G1, B1, R2, G2, B2):** Bu pinler, ekranda görüntülenecek renk verilerini (kırmızı, yeşil, mavi) taşır. Genellikle R1, G1, B1 pinleri panelin üst yarısını, R2, G2, B2 pinleri ise alt yarısını kontrol eder, bu da paralel veri aktarımına olanak tanır.
2. **Satır Adresleme Pinleri (A, B, C, D, E):** Bu pinler, veri gönderilecek olan satırı seçmek için kullanılır. P4 panellerde yaygın olarak 1/16 tarama oranı kullanıldığından, genellikle A, B, C ve D pinleri kullanılırken, daha yüksek çözünürlüklü paneller E pinini de gerektirebilir.
3. **Kontrol Pinleri (CLK, LAT, OE):**
    - **CLK (Saat):** Veri bitlerinin panelin sürücü entegrelerine (shift registers) aktarılmasını senkronize eden yüksek hızlı bir saat sinyalidir.
    - **LAT (Mandal/Latch):** Bir satırın tüm veri bitleri panele yüklendiğinde, bu sinyal verinin LED'lere aktarılmasını sağlar.
    - **OE (Çıkış Etkinleştirme):** Paneldeki LED'lerin parlaklığını kontrol etmek için kullanılır ve genellikle bir PWM (Darbe Genişliği Modülasyonu) sinyali ile sürülür.

STM32 ile Haberleşme Yöntemleri

STM32 gibi yüksek performanslı bir mikrokontrolcü, P4 paneli sürmek için farklı yöntemler kullanabilir. Panelin sürekli ve yüksek hızlı bir şekilde yenilenmesi gerektiğinden, veri aktarımı işlemci üzerinde önemli bir yük oluşturur. Bu nedenle, en verimli yöntemler, işlemcinin yükünü azaltan donanım hızlandırmalarını içerir.

- **Bit-Banging:** En basit yöntemdir ancak en az verimli olanıdır. İşlemci, her bir pini tek tek kontrol ederek veri ve kontrol sinyallerini manuel olarak üretir. Bu yöntem, yüksek çözünürlüklü ve yüksek yenileme hızlı paneller için genellikle yetersizdir ve görüntüde titremeye neden olabilir.
- **DMA (Doğrudan Bellek Erişimi):** Bu, STM32'nin en önemli özelliklerinden biridir. DMA, işlemci müdahalesi olmadan bellekteki verileri doğrudan bir çevresel birime (bu durumda GPIO pinlerine) aktarabilir. Bu yöntem, sürekli veri akışını sağlarken işlemciyi serbest bırakır ve daha karmaşık görevler için kullanılabilmesini sağlar.

Birçok özel HUB75 kütüphanesi, paneli verimli bir şekilde sürmek için DMA'yı kullanır.

- **SPI/I2S Çevresel Birimleri:** Bazı kütüphaneler, veri aktarımı için SPI (Seri Çevresel Arayüz) gibi özel donanım çevresel birimlerini kullanır. Bu yaklaşım, bit-banging yöntemine göre daha hızlı ve daha güvenilirdir. Ancak, HUB75 protokolünün doğası gereği, tüm pinler tek bir standart protokol altında sürülemez; bu nedenle genellikle veri pinleri SPI ile, diğer kontrol pinleri ise GPIO ile kontrol edilir.
- **Özel Yazılım Kütüphaneleri:** HUB75 panellerini sürmek için STM32'ye özgü optimize edilmiş kütüphaneler mevcuttur. Bu kütüphaneler, paneli başlatma, parlaklık ve renk ayarlarını yapma, görüntü verilerini gönderme ve yenileme işlemlerini kolaylaştıran yüksek seviyeli işlevler sunar. Bu kütüphaneler, genellikle arka planda DMA gibi donanım hızlandırmalarını kullanarak işlemci yükünü minimumda tutar. Örneğin,

  `kostaman/HUB75` ve `Thakaa/HUB75E-Driver` gibi kütüphaneler, STM32 için geliştirilmiş popüler çözümlerdir.

Neden r1 b1 g1 ve r2 b2 g2

R1, G1, B1 ve R2, G2, B2 veri pinlerinin varlığı, LED panelin çalışma prensibinin temel bir parçası olan dahili çoklama (multiplexing) ve yarı panel sürüşü (half-panel driving) mimarisine dayanır. Bu yapı, yüksek çözünürlüklü görüntülerin daha az sayıda pinle ve daha hızlı bir şekilde ekrana aktarılmasını sağlar.

Yarı Panel Sürüşü (Half-Panel Driving)

Bir LED paneli sürmenin en verimli yollarından biri, paneli yatay olarak ikiye bölmektir. Bu sayede mikrokontrolcü aynı anda panelin hem üst yarısını hem de alt yarısını güncelleyebilir. R1, G1 ve B1 pinleri panelin üst yarısındaki piksellere renk verilerini iletirken, R2, G2 ve B2 pinleri alt yarıdaki piksellere veri gönderir.

Bu mimarinin temel amacı, verimliliği artırmaktır. Örneğin, 32 satırlık bir panelde, 1'den 16'ya kadar olan satırlar üst yarıyı, 17'den 32'ye kadar olan satırlar ise alt yarıyı oluşturur. STM32 her seferinde bir satır adresleme pini (A, B, C, D) gönderdiğinde, bu pin hem üst yarıdaki bir satırı (örneğin satır 1) hem de alt yarıdaki ilgili satırı (satır 17) seçer. Ardından, R1, G1, B1 pinleri aracılığıyla satır 1'e ait renk verileri ve R2, G2, B2 pinleri aracılığıyla satır 17'ye ait renk verileri eş zamanlı olarak gönderilir.

Bu yöntem sayesinde, 32 satırlık bir paneli sürmek için gereken döngü sayısı 32'den 16'ya düşürülür. Bu, yenileme hızını iki katına çıkararak ekranda oluşabilecek titreme (flickering) sorunlarını önemli ölçüde azaltır ve daha akıcı bir görüntü elde edilmesini sağlar.

Sinyaller ve Pin Fonksiyonları

Veri pinlerinin bu şekilde ikiye ayrılması, HUB75 protokolünün çalışmasını sağlayan diğer kritik sinyallerle birlikte işler:

- **R1, G1, B1 & R2, G2, B2 (Veri Pinleri):** Paneldeki her pikselin rengini (Kırmızı, Yeşil, Mavi) belirlemek için kullanılır. Bu pinler, bir kaydırma yazmacına (shift register) veri aktarımı için kullanılır.
- **CLK (Saat):** Mikrokontrolcü, her bir bitin veri pinleri aracılığıyla kaydırma yazmacına aktarılması için CLK pinine bir saat sinyali gönderir.
- **LAT (Mandal/Latch):** Bir satıra ait tüm veriler (hem R1, G1, B1 hem de R2, G2, B2 pinlerinden gelen veriler) kaydırma yazmacına yüklendiğinde, LAT pini bir sinyal göndererek bu verinin LED'lere kopyalanmasını sağlar.
- **OE (Çıkış Etkinleştirme):** Bu pin, paneli tamamen açıp kapatmaya yarayan bir anahtar görevi görür ve genellikle bir PWM (Darbe Genişliği Modülasyonu) sinyali ile sürülerek genel ekran parlaklığı ayarlanır.

Sonuç olarak, R1, G1, B1 ve R2, G2, B2 pinlerinin ayrı ayrı bulunmasının temel sebebi, paneli mantıksal olarak ikiye bölerek veri aktarımını hızlandırmak ve görüntü yenileme oranını artırmaktır. Bu durum, daha akıcı video oynatımı ve titremesiz bir görsel deneyim için kritik bir mühendislik çözümüdür.

Pinler ve pikseller arasındaki bağlantı

LED matris panelleri, her pikselin tek tek kontrol edildiği bir sistem yerine, satır ve sütunlardan oluşan bir matris yapısı kullanır. Bu yapı, binlerce LED'i sürmek için gereken pin sayısını önemli ölçüde azaltır. Panellerdeki her piksel bir LED kümesidir ve her küme bir kırmızı (R), bir yeşil (G) ve bir mavi (B) LED'den oluşur.

Yarı Panel Sürüşü (Half-Panel Driving) Mimarisi

HUB75 arayüzüne sahip paneller, özellikle daha büyük çözünürlükler için, paneli yatay olarak iki mantıksal bölüme ayırır: üst yarı ve alt yarı. Bu teknik, "yarı panel sürüşü" olarak bilinir ve veri aktarımını hızlandırmak için kullanılır.

- **R1, G1, B1 Pinleri:** Bu pinler panelin **üst yarısındaki** pikseller için renk verilerini taşır.
- **R2, G2, B2 Pinleri:** Bu pinler ise panelin **alt yarısındaki** pikseller için renk verilerini taşır.

Bu sayede, kontrolcü tek bir döngüde iki satırı birden güncelleyebilir. Örneğin, 32 satırlık bir panelde, 1'den 16'ya kadar olan satırlar üst yarıyı, 17'den 32'ye kadar olan satırlar ise alt yarıyı oluşturur. Mikrokontrolcü, A, B, C, D adresleme pinlerini kullanarak aynı anda hem üst yarıdaki bir satırı (örneğin satır 1) hem de alt yarıdaki ilgili satırı (satır 17) seçer. Bu seçimin ardından, R1, G1, B1 pinleri ile satır 1'e ait veriler ve R2, G2, B2 pinleri ile de satır 17'ye ait veriler **eş zamanlı olarak** panele gönderilir.

Pinler ve Piksel Bağlantısı

Bu veri pinleri, panelin içinde her satıra karşılık gelen kaydırma yazmaçlarına (shift registers) bağlıdır. İşlemci veri pinlerine (R1, R2, G1, G2, B1, B2) piksel verisini yüklerken, CLK (saat) sinyaliyle bu verinin kaydırma yazmaçlarında ilerlemesini sağlar. Panelin bir satırı için tüm

piksel verisi yüklendikten sonra, LAT (mandal) sinyali ile bu veriler, seçilen satırdaki LED'lere aktarılır. OE (Çıkış Etkinleştirme) pini ise tüm panelin parlaklığını kontrol etmek için kullanılır ve istenmeyen piksellenmeyi önlemek amacıyla verinin panale kilitlenmesi tamamlanana kadar paneli kapalı tutar.

Bu yöntem, paneli satır satır tarayarak görüntüyü yenileme hızını artırır ve daha akıcı bir görüntü elde edilmesini sağlar. Bu, özellikle video veya animasyon gibi hareketli içeriklerde titreme (flickering) sorununu minimize eder. R1, G1, B1 ve R2, G2, B2 pinleri arasındaki bu mimari fark, panellerin daha karmaşık sürücü devrelerine sahip olmasına ve daha verimli çalışmasına olanak tanır.

Display p4 adresleme pinleri ve data pinleri arasındaki bağlantı ve etkileşim

Adresleme (A, B, C, D) ve veri (R1, G1, B1, R2, G2, B2) pinleri arasındaki etkileşim, P4 gibi LED ekranların görüntüyü nasıl oluşturduğunu belirleyen temel bir mekanizmadır. Bu etkileşim, bir matris tarama (multiplexing) tekniğine dayanır ve tek tek pikselleri adreslemek yerine satır bazında veri göndermeyi içerir .

Etkileşimin Temel Prensibi

STM32 gibi bir mikrodenetleyicinin P4 paneli sürmesi, sürekli ve çok hızlı bir döngüde gerçekleşen bir dizi adımdan oluşur . Bu süreç, "tarama (scanning)" olarak adlandırılır ve görüntünün tamamını göstermek için her satırı art arda yakar.

1. **Satır Seçimi (Adresleme):** İlk olarak, mikrokontrolcü A, B, C ve D adresleme pinlerine belirli bir ikili kod (binary) göndererek paneldeki hangi satırın etkinleştirileceğini seçer. P4 panellerde yaygın olarak 1/16 tarama oranı kullanılır, bu da 16 satırlık bir tarama döngüsü olduğu anlamına gelir. Bu döngü, A, B, C, D pinleri ile 0000'dan 1111'e (ikili) kadar adresleme yaparak her satırı tek tek seçer .
2. **Veri Gönderme:** Adresleme pinleri bir satırı seçtiğinde, veri pinleri (R1, G1, B1, R2, G2, B2) o satırdaki tüm piksellerin renk verilerini panele göndermeye başlar. Bu veri, panelin içindeki kaydırma yazmaçlarına (shift registers) yüklenir. Bu işlem, CLK (saat) pini aracılığıyla senkronize edilir. Her bir saat sinyali ile bir bit verisi kaydırma yazmacında bir sonraki konuma kayar.
3. **Veri Kilitleme ve Görüntüleme:** Satırın tüm piksel verisi (kaydırma yazmaçları tam olarak dolduğunda) gönderildikten sonra, LAT (mandal/latch) pinine bir sinyal gönderilir. Bu sinyal, kaydırma yazmaçlarındaki verinin, adresleme pinleri tarafından seçilen satırdaki LED'lere kopyalanmasını sağlar.
4. **Ekranı Etkinleştirme ve Devam Etme:** Bu sırada, OE (çıkış etkinleştirme) pini genellikle kapalı (yüksek) tutulur, bu da görüntüleme sırasında herhangi bir titremeyi veya bozulmayı önlemek için LED'lerin kapalı kalmasını sağlar. Veri kilitleme işlemi tamamlandıktan sonra OE pini açılır (düşük), bu da yeni verinin ekranda görünmesini sağlar. Tüm bu süreç, her satır için inanılmaz bir hızla tekrar edilerek gözümüzün görüntüdeki titremeyi algılamasını engeller ve sabit, akıcı bir görüntü yaratır .

Verimlilik İçin Yarı Panel Mimarisi

Bu sürecin verimli bir şekilde çalışabilmesi için P4 paneller genellikle "yarı panel sürüşü" (half-panel driving) mimarisine sahiptir. Bu mimaride, panel yatay olarak ikiye bölünür (üst ve alt yarı) ve her döngüde iki satır eş zamanlı olarak adreslenir.

- **Adresleme Pinleri (A, B, C, D):** Tek bir adresleme pini kombinasyonu (örneğin A=0, B=0, C=0, D=0), panelin hem üst yarısındaki satır 1'i hem de alt yarısındaki satır 17'yi seçer.
- **Veri Pinleri (R1, G1, B1 & R2, G2, B2):** Bu eş zamanlı adresleme sayesinde, R1, G1, B1 pinleri ile üst satıra (satır 1) ait veriler, R2, G2, B2 pinleri ile ise alt satıra (satır 17) ait veriler aynı anda gönderilir.

Bu iki paralel veri hattı sayesinde, bir P4 paneldeki 32 satırın tamamını taramak için gereken süre yarıya iner . Bu durum, daha yüksek yenileme hızı ve daha akıcı video performansı sağlar, özellikle yüksek çözünürlüklü P4 gibi panellerde . Bu sistem, adresleme ve veri pinlerinin senkronize çalışarak karmaşık görüntü verilerini dinamik olarak bir matris üzerinde görüntülemesini sağlayan zarif bir donanım/yazılım etkileşiminin sonucudur

320x160 boyutunda bir p4 displaye stm32 ile yazmak

STM32 ile 320x160 piksel boyutunda bir P4 ekranı sürmek, dikkatli bir donanım kurulumu ve optimize edilmiş bir yazılım yaklaşımı gerektiren karmaşık bir projedir. Bu boyuttaki bir ekran, 80x40 piksel çözünürlüğe sahip dört adet 320x160 mm P4 modülünün birleştirilmesiyle elde edilebilir. Bu, yüksek çözünürlüklü görüntüleme için güçlü bir temel oluşturur ancak işlemci ve RAM üzerinde önemli bir yük yaratır. İşte bu projeyi başarılı bir şekilde tamamlamak için adım adım bir rehber:

1. Donanım Gereksinimleri ve Entegrasyonu

Yüksek çözünürlüklü bir P4 ekranı verimli bir şekilde sürmek için, sadece STM32'yi panele bağlamak yeterli değildir. Projenin kararlı çalışmasını sağlamak için bazı ek donanım bileşenleri kritik öneme sahiptir.

- **Güç Kaynağı:** Bir P4 panel, tam parlaklıkta çalışırken önemli miktarda akım (ortalama 8-10A) çekebilir. Bu nedenle, STM32'nin USB portu veya dahili regülatörü gibi kaynaklar yetersiz kalacaktır. 320x160 mm boyutundaki bir P4 modülünün maksimum güç tüketimi 13W'a kadar çıkabilirken , birden fazla modül kullanılması durumunda bu değer katlanarak artar. Projeninize güç vermek için, paneli doğrudan besleyecek ve yeterli akım kapasitesine sahip (örneğin, 5V, 10A veya üzeri) harici bir güç kaynağı kullanmak zorunludur.
- **Seviye Dönüştürücü (Level Shifter):** STM32 mikrodenetleyicileri genellikle 3.3V mantık seviyesinde çalışır. Ancak, P4 panelleri kontrol eden HUB75 arayüzü ve sürücü entegreleri 5V mantık seviyesi bekler. Bu iki voltaj seviyesi arasındaki uyumsuzluk, doğrudan bağlantı halinde STM32'nin GPIO pinlerine zarar verebilir. Bu sorunu çözmek için, STM32'den panele giden tüm veri, adres ve kontrol sinyal hatlarına 3.3V'dan 5V'a seviye dönüştürme işlemi yapan bir entegre (örneğin, 74HC245 gibi) eklemek gereklidir.
- **HUB75 Kablolama:** 320x160mm boyutundaki bir P4 modülü, standart bir HUB75 soketine sahiptir. Panelleri birbirine bağlamak için genellikle paketle birlikte gelen düz

veri kabloları kullanılır. Fiziksel bağlantı, bu kabloları panellerin giriş ve çıkış soketlerine takarak kolayca yapılır. Birden fazla paneli zincirleme (daisy-chain) yöntemiyle bağlayarak 320x160mm boyutunda tek bir büyük sanal ekran oluşturabilirsiniz.

2. Yazılım Mimarisi ve Kodlama Yöntemleri

Yazılım tarafında, P4 panelin yüksek çözünürlüğünü ve yenileme hızını desteklemek için optimize edilmiş bir yaklaşım benimsemek gereklidir. Basit bit-banging yöntemleri, bu panellerin performansını tam olarak kullanamayacağı için önerilmez.

- **Kütüphane Seçimi:** HUB75 panellerini sürmek için STM32'ye yönelik özel kütüphaneler mevcuttur. Bu kütüphaneler, panelin yüksek hızlı tarama ve yenileme işlemlerini kolaylaştıran yüksek seviyeli işlevler sunar. STM32CubeIDE gibi araçlarla kolayca entegre edilebilen

  `kostaman/HUB75` veya

  `Thakaa/HUB75E-Driver` gibi kütüphaneler, bu projeniz için uygun başlangıç noktaları olabilir. Bu kütüphaneler, genellikle DMA (Doğrudan Bellek Erişimi) özelliğini kullanarak işlemci üzerindeki yükü minimize eder.

- **DMA ve Zamanlayıcı (Timer) Kullanımı:** Görüntünün ekrana titreme olmadan aktarılabilmesi için, STM32'nin DMA ve Timer çevresel birimleri kullanılmalıdır. DMA, görüntü tamponundaki (frame buffer) veriyi işlemci müdahalesi olmadan doğrudan GPIO pinlerine aktarırken, bir zamanlayıcı bu veri aktarımını belirli aralıklarla tetikleyerek ekranın sürekli yenilenmesini sağlar. Bu mimari, işlemcinin diğer görevler için serbest kalmasına olanak tanır ve akıcı bir görüntü performansı sunar.
- **Görüntü Tamponu (Frame Buffer):** SCADA ekranınızda göstereceğiniz grafikler ve veriler, öncelikle bir görüntü tamponu (RAM'de ayrılmış bir alan) içinde hazırlanmalıdır. Bu tampon, paneli süren DMA tarafından okunacak olan piksel verilerini içerir. Kütüphaneler, genellikle bu tamponu yönetmek için

  `display.drawPixel(x, y, color)` gibi yüksek seviyeli fonksiyonlar sunar. Bu sayede, karmaşık görüntü işleme algoritmaları yerine, sadece piksel verisini bu tampona yazarak kolayca SCADA ekranınızı oluşturabilirsiniz.

3. Yazılım Geliştirme Adımları

Proje geliştirme süreci aşağıdaki adımlarla ilerleyebilir:

1. **STM32CubeIDE Projesi Oluşturma:** İlk olarak, STM32CubeIDE'de yeni bir proje başlatın. Doğru STM32 serisini ve mikrodenetleyici modelini seçtiğinizden emin olun.
2. **Periferik Konfigürasyonu:** Proje konfigürasyon aracında, paneli kontrol etmek için kullanacağınız GPIO pinlerini (Veri, Adres, Kontrol) Output olarak ayarlayın. Görüntü yenilemesi için DMA ve Timer birimlerini de doğru şekilde yapılandırın. Bu, kütüphanenin beklediği HAL fonksiyonlarını oluşturacaktır.

3. **Kütüphane Entegrasyonu:** Seçtiğiniz HUB75 kütüphanesini (örneğin `kostaman/HUB75`) projenize ekleyin. Kütüphanenin kaynak dosyalarını projeye dahil edin.
4. **Kod Geliştirme:**
    - `main()` fonksiyonunda kütüphaneyi başlatın ve ekran boyutlarını (bu durumda 320x160mm, yani 80x40 piksel çözünürlüğe denk gelir) ve tarama oranını (P4 paneller için genellikle 1/16) doğru şekilde tanımlayın.
    - SCADA verilerini (örneğin ADC okumaları, sensör verileri) okumak için ilgili çevresel birimleri (örneğin ADC, UART) yapılandırın.
    - Ana `while(1)` döngüsü içinde, okunan verilere göre SCADA arayüzünü görüntü tamponuna çizin. Bu aşamada, `drawPixel`, `drawCircle`, `drawString` gibi fonksiyonlar kullanılabilir.
    - Paneli sürekli yenilemek için `while(1)` döngüsünde kütüphanenin `displayBufferPixels()` gibi fonksiyonunu çağırın.