

Unidade didáctica

Desenvolvimento e construción de prototipos electrónicos e Sistemas de telecomunicación e informática.

Familia profesional	Electricidade e electrónica
Ciclo formativo	Desenvolvimento de produtos electrónicos
Grao	Superior
Módulos	Desenvolvimento e construción de prototipos electrónicos Desenvolvimento de proxectos de produtos electrónicos
Unidade didáctica	Programación de Pic's en Linguaxe C baixo protocolos SPI e I ² c
Actividade	Construción dun programador e programación para o Pic 18F45J10 baixo C con protocolos SPI e I²C
Autores	Grupo de traballo de profesores de electrónica do IES Ricardo Mella e do IES de Tomiño

1.	Datos da actividade.....	4
1.1	Titulo e descrición.....	4
1.2	Obxectivos.....	4
1.3	Contidos	6
1.3.1	Contidos procedementais.....	6
	Desenvolvemento dun proxecto de aplicación electrónica:	6
	Realización da posta a punto e da documentación dun proxecto electrónico:	6
	Elaboración da planificación e xestión dun proxecto electrónico:	7
	Realización dun proxecto de aplicación electrónica no ámbito industrial, no que interveñan técnicas de medida, control, regulación e potencia electrónica. Incluirá tecnoloxías analóxicas, dixitais e microprogramables:	7
	Realización dun proxecto de aplicación electrónica no ámbito das telecomunicacións, no que interveñan distintas técnicas de modulación e de transmisión en distintos soportes. Incluirá tecnoloxías analóxicas, dixitais e microprogramables:	7
	Elaboración de esquemas electrónicos con soporte informático	8
	Deseño de circuitos impresos con soporte informático	8
	Confección da documentación técnica para a construción de placas de circuito impreso e montaxe de prototipos	8
	Construción de circuitos impresos para prototipos.....	9
	Realización da montaxe de compoñentes electrónicos nas placas de circuito impreso	9
	Selección e aplicación de probas de funcionalidade nos prototipos realizados	9
	Realización de probas de fiabilidade nos prototipos realizados	9
1.3.2	Contidos conceptuais	10
	Fases no deseño e desenvolvemento dun proxecto electrónico:	10
	Posta a punto e documentación:	10
	Planificación e xestión de proxectos:.....	10
	Aspectos organizativos	10
	Técnicas de deseño de circuitos impresos.....	11
	Construción de placas de circuito impreso	11
	Montaxe de compoñentes nas placas de circuito impreso	11
	Probas funcionais e axustes	11
	Control de calidade e fiabilidade.....	11
1.3.3	Contidos actitudinais	12
	Traballo en grupo.....	12
	Comportamento interpersonal	12
	Optimización do traballo na realización de tarefas	12
	Realización estruturada do traballo na execución de tarefas	12
	Compromiso coas a brigas asociadas ó traballo	12
	Avaliación de resultados	12
	Coordinación dos recursos humanos	13
1.3.4	Dominio profesional.....	13
	Medios de produción ou tratamento da información.....	13
	Materiais e produtos intermedios.....	13
	Principais resultados do traballo (produtos e/ou servizos)	13
	Procesos, métodos e procedementos	14
	Información (natureza, tipo e soportes)	14
	Persoal e/ou organizacións destinatarias	14
1.4	Aspectos metodolóxicos	14
1.4.1	Requisitos mínimos de espazos e instalacións	14

2.	Programación de Pic's en Linguaxe C baixo protocolos SPI e I²c	16
2.1	Introdución	16
2.2	Programas Feitos co Relator en Linguaxe C baixo protocolos I ² C e SPI.....	16
2.2.1	PARPADEO DE LED VERDE POR SEGUNDO.	18
2.2.2	PARPADEO DE TRES LEDS SECUENCIAL VARIANDO O TEMPO	19
2.2.3	RELOXO CONTADOR DE SEGUNDOS EN FORMATO SSSS.....	21
2.2.4	RELOXO DE MINUTOS E SEGUNDOS FORMATO MM SS	25
2.2.5	RELOXO PROGRAMABLE POR BOTONS EN FORMATO MM SS.....	28
2.2.6	LETURA DE TEMPERATURA BAIXO I ² C EN LCD E DISPLAY	36
2.3	Construción dun Programador de PIC's co OrCad	40
2.3.1	Previo o deseño co programa OrCad.....	40
2.3.2	Deseño co programa OrCad Capture. Placa do PIC	41
2.3.3	Deseño co programa OrCad Layout. Placa do PIC	42
2.3.4	Deseño co programa OrCad Capture. Placa dos Displays e Rexistros de Desprazamento.....	46
2.3.5	Deseño co programa OrCad Layout. Placa dos Displays e Rexistros de Desprazamento.....	47
2.3.6	Deseño co programa OrCad Capture. Placa do LCD	51
2.3.7	Deseño co programa OrCad Layout. Placa do LCD	52
2.3.8	Deseño co programa OrCad Capture. Placa dos LED e Interruptores	55
2.3.9	Deseño co programa OrCad Layout. Placa dos LED e Interruptores	56
2.3.10	Imaxes do Programador do PIC, Placa PIC, Displays e Rexistros de Desprazamento e conectores utilizados para comunicar as placas	59
2.3.11	Anexo	62

1. Datos da actividade

1.1 Titulo e descrición

Título: Construción dun programador e programación para o Pic18F45J10 baixo C e protocolos SPI e I²C con periféricos de entrada e saída.

Descrición: Elaboración de PCB's co ORCAD, con edición de librerías e footprints. Programación da me utilizando protocolos I²C e SPI.

1.2 Obxectivos

DESENVOLVEMENTO DE PROXECTOS DE PRODUTOS ELECTRÓNICOS

- Os algoritmos e diagramas de fluxo da solución programada reflicten adecuadamente o tratamento dos datos, a secuencia e o fluxo de información ó longo dos programas.
- As técnicas utilizadas no deseño dos programas teñen en conta a programación modular e as estruturas de control básicas da programación estruturada.
- Realízase a elección da linguaxe de programación (de alto e/ou baixo nivel) en función as prestacións de velocidade, a portabilidade e as ferramentas de desenvolvemento dispoñibles.
- Codifícanse convenientemente os algoritmos da solución adoptada, utilizando as estruturas de control e recursos da linguaxe (ou linguaxes) seleccionados.
- Os programas da aplicación inclúen rutinas e procedementos estándar incluídos en librerías previamente normalizadas.
- O código dos programas está suficientemente comentado, garantindo un posterior mantemento.
- As probas funcionais do programa aseguran que o tratamento dos datos se axusta ó especificado nos correspondentes diagramas de fluxo.
- As probas conxuntas do software e do hardware da aplicación aseguran o cumprimento das especificacións funcionais e prestacións da aplicación.
- Realízase a documentación dos programas (diagramas de fluxo, listaxes de código,...) coa calidade prescrita e no soporte e formato normalizados.

DESENVOLVEMENTO E CONSTRUCIÓN DE PROTOTIPOS ELECTRÓNICOS

- Describir a tipoloxía e características dos programas informáticos usados para o debuxo de esquemas electrónicos, seleccionado os parámetros que os configuran para un uso axeitado deles.
- Realizar a selección de compoñentes e os seus atributos nas librerías, creándoos se non existen e situándoos no formato elixido a partir das especificacións dos circuítos.
- Confeccionar esquemas de aplicacións electrónicas realizando a interconexión dos compoñentes, seguindo procedementos normalizados en debuxo de circuítos.
- Elaborar bases de datos que conteñan a documentación dos esquemas realizados, indicando as características técnicas e funcionais de cada un.

- Describir a tipoloxía e características dos programas informáticos usados para o deseño de circuítos impresos, seleccionando os parámetros que os configuran para un uso correcto deles.
- Precisar as características funcionais requiridas polo deseño de circuítos impresos, seleccionando os compoñentes e os seus atributos nas librerías, ou creándoos se é preciso, situándoos no formato elixido a partir das especificacións dos circuítos.
- Definir a forma e tamaño de placas de circuítos impresos, situando os compoñentes con criterios de optimización do espazo, seguridade térmica, eléctrica e electromagnética.
- Aplicar as técnicas de trazado manual e/ou automático de pistas de circuítos impresos, dunha ou dúas caras, entre os diferentes compoñentes a partir dos listados de conexións.
- Elaborar bases de datos que conteñan a documentación das placas realizadas, indicando as características técnicas e funcionais de cada unha.
- Realizar as copias impresas dos diferentes circuítos, placas, listados de compoñentes e das diferentes fases da realización dos prototipos.
- Confeccionar a documentación técnica de construción e montaxe de circuítos electrónicos en formato normalizado, utilizando as ferramentas informáticas de deseño de forma que responda ás diferentes fases da construción de prototipos.
- Diferenciar as placas para a montaxe de compoñentes de inserción e de montaxe superficial, identificando os tipos de dieléctrico e seleccionando os axentes de revelado, gravado e decapado para usar no proceso de construción de circuítos.
- Executar os procesos de tradeado manual e/ou automatizado, metalizado de trades, fotosensibilización, revelado, gravado, decapado e serigrafado das placas de circuítos impresos, seguindo métodos normalizados e baixo as normas de seguridade.
- Vencellar os diferentes procesos de montaxe manual e automático de compoñentes de inserción, compoñentes superficiais e técnicas de soldadura manual e automática, coas máquinas, ferramentas e secuencias de cada operación.
- Realizar a montaxe manual de placas de circuítos impresos, definindo o procedemento para seguir de acordo coa documentación técnica, utilizando as ferramentas e os materiais axeitados.
- Confeccionar placas de circuítos impresos de forma automatizada, programando a máquina de montaxe de compoñentes seguindo as súas instrucións, elixindo o proceso de soldadura en función dos compoñentes e do tipo de placa utilizados.
- Diferenciar as características de análise estática e análise funcional dun prototipo electrónico, establecendo o tipo de alimentación eléctrica para o seu correcto funcionamento.
- Realizar as conexións dos prototipos verificando o seu funcionamento, aplicando procedementos de proba e axuste establecidos na documentación técnica.
- Precisar e aplicar procedementos axeitados de control de calidade a partir da revisión dos compoñentes e materiais, do proceso de montaxe e do prototipo final.
- Establecer e pór en práctica procedementos axeitados de control de fiabilidade a partir das análises térmicas, eléctricas, mecánicas, de humidade e da vida do produto.
- Elaborar informes-memoria das actividades desenvolvidas, dos resultados obtidos e dos medios utilizados, estruturándoas nos apartados precisos para unha adecuada documentación.

1.3 Contidos

1.3.1 Contidos procedementais

DESENVOLVEMENTO DE PROXECTOS DE PRODUTOS ELECTRÓNICOS

Desenvolvemento dun proxecto de aplicación electrónica:

- Especificacións técnicas. Elaboración coa precisión requirida.
- Información técnica comercial ou en documentación doutros proxectos:
 - Selección da necesaria como fonte de información.
 - Confección de arquivos para o desenvolvemento da aplicación.
- Diagramas de bloques. Elaboración partindo de circuítos similares para a resolución a nivel funcional da aplicación.
- Compoñentes electrónicos. Realización dos cálculos técnicos e matemáticos necesarios para a súa selección.
- Circuítos electrónicos analóxicos e dixitais:
 - Selección das tecnoloxías e dos compoñentes máis adecuadas, partindo dos manuais e a documentación técnica.
 - Elaboración dos esquemas.
 - Simulación por medios informáticos.
- Maquetas electrónicas:
 - Construción seleccionando o procedemento máis adecuado.
 - Uso de técnicas de montaxe rápida.
- Dispositivos microprogramables utilizados na aplicación:
 - Selección da linguaxe para a elaboración do programa de control.
 - Confección dos programas, cos oportunos comentarios para o seu correcto mantemento.

Realización da posta a punto e da documentación dun proxecto electrónico:

- Probas funcionais:
 - Determinación das medidas que hai que realizar, indicando os parámetros máis críticos.
 - Selección da instrumentación e os equipos de medida para o axuste do proxecto, tendo en conta as probas de fiabilidade que debe cumprilo produto.
 - Realización das medidas e os axustes adecuados.
- Documentación:
 - Elaboración dos formatos onde se deben recollelos resultados das medidas, probas e ensaios que se deben realizar no prototipo.
 - Selección e organización da documentación base correspondente á aplicación.
 - Determinación da ferramenta informática máis adecuada ás características e o tipo de documentación que se vai elaborar.
 - Realización da documentación técnica.

Elaboración da planificación e xestión dun proxecto electrónico:

- Normativa técnica e administrativa. Selección da que atinxa ó proxecto e que poida afectala súa homologación.
- Fases de desenvolvemento do proxecto. Organización, secuenciando as tarefas realizadas en cada unha delas.
- Viabilidade do produto. Realización dun estudio pormenorizado.
- Recursos materiais, humanos e de infraestrutura. Determinación dos necesarios para o desenvolvemento do proxecto.
- Técnicas e gráficas (PERT, GANTT). Utilización para reflectilas actividades, sucesos e puntos críticos no desenvolvemento do proxecto.

Realización dun proxecto de aplicación electrónica no ámbito industrial, no que interveñan técnicas de medida, control, regulación e potencia electrónica. Incluirá tecnoloxías analóxicas, dixitais e microprogramables:

- Especificacións funcionais e de calidade:
 - Selección dos tipos de tecnoloxías, dispositivos e materiais, procesos de fabricación, ferramentas, equipos e máquinas que se deben empregar.
 - Análise de viabilidade da aplicación proposta.
 - Planificación do proxecto nas súas distintas fases.
 - Elaboración de esquemas e planos necesarios para a construción do prototipo.
 - Confección da memoria descritiva de funcionamento dos circuítos.
 - Creación da listaxe de materiais e fontes subministradoras.
 - Construción da maqueta correspondente.
 - Elaboración dos programas de control na linguaxe e cos formatos estándar requiridos.
 - Realización de probas funcionais, de calidade e de fiabilidade precisas.
 - Confección do orzamento correspondente.

Realización dun proxecto de aplicación electrónica no ámbito das telecomunicacións, no que interveñan distintas técnicas de modulación e de transmisión en distintos soportes. Incluirá tecnoloxías analóxicas, dixitais e microprogramables:

- Especificacións funcionais e de calidade:
 - Selección dos tipos de tecnoloxías, dispositivos e materiais, procesos de fabricación, ferramentas, equipos e máquinas que se deben empregar.
 - Análise de viabilidade da aplicación proposta.
 - Planificación do proxecto nas súas distintas fases.
 - Elaboración de esquemas e planos necesarios para a construción do prototipo.
 - Confección da memoria descritiva de funcionamento dos circuítos.
 - Creación da listaxe de materiais e fontes subministradoras.
 - Construción da maqueta correspondente.
 - Elaboración dos programas de control na linguaxe e cos formatos estándar requiridos.
 - Realización de probas funcionais, de calidade e de fiabilidade precisas.

- Confección do orzamento correspondente.

DESENVOLVEMENTO E CONSTRUCCIÓN DE PROTOTIPOS ELECTRÓNICOS

Elaboración de esquemas electrónicos con soporte informático

- Aplicacións informáticas usadas na creación de esquemas electrónicos. Selección, obtendo a súa tipoloxía e características.
- Componentes requiridos. Selección nas librerías do programa ou creación se non existen, editando os seus atributos.
- Interconexións entre componentes. Realización seguindo procedementos normalizados en e listados de conexións.
- Documentación do circuíto.
 - Configuración de arquivos con listas de material e anotacións.
 - Obtención de copias impresas a través dos periféricos informáticos.

Deseño de circuitos impresos con soporte informático

- Características funcionais e técnicas das placas. Definición segundo as diferentes necesidades.
- Aplicacións informáticas usadas na creación de esquemas electrónicos. Selección, obtendo a súa tipoloxía e características.
- Componentes requiridos.
 - Selección nas librerías do programa ou creación, se non existen.
 - Edición dos atributos.
 - Definición do tamaño e forma da placa do circuíto impreso.
- Técnicas de trazado de pistas.
 - Definición segundo as especificacións realizadas.
 - Realización do trazado manual e/ou automático de pistas entre os diferentes componentes, a partir da listas de conexións, situando os componentes de forma óptima e tendo en conta as medidas de seguridade térmica electromagnética.
- Documentación do circuíto.
 - Configuración de arquivos con listas de material e anotacións.
 - Obtención de copias impresas a través dos periféricos informáticos.

Confección da documentación técnica para a construción de placas de circuíto impreso e montaxe de prototipos

- Esquemas eléctrico/electrónico. Confección no formato normalizado.
- Listados de materiais precisos. Realización agrupándoos de acordo coa súa tipoloxía, funcionalidade e características.
- Listados de conexións. Elaboración para uso como elemento de comprobación.
- Documentación de placas de circuíto impreso. Confección en formato normalizado e compostas polas máscaras de soldadura, máscaras de pistas e planos de tradeado.
- Planos necesarios para a montaxe dos componentes dos circuitos. Realización de forma que responda ás distintas fases da construción dos prototipos.

Construción de circuítos impresos para prototipos

- Técnicas necesarias.
 - Clasificación das placas en función do tipo de dieléctrico e do tipo de montaxe.
 - Confección do método de actuación.
 - Selección de útiles e produtos adecuados á realización.
 - Selección dos materiais para empregar na realización de placas.
 - Realización de cada un dos procesos con meticulosidade, seguindo as normas de seguridade.
- Informe-memoria da construción de placas de circuítos impresos. Elaboración, estruturando os resultados obtidos para unha adecuada documentación.

Realización da montaxe de compoñentes electrónicos nas placas de circuítos impresos

- Técnicas de montaxe:
 - Definición do proceso para seguir, segundo a documentación técnica.
 - Selección das ferramentas e materiais axeitados.
 - Realización da montaxe dos compoñentes, seguindo procedementos establecidos e baixo as normas de seguridade.
 - Inspección da placa montada para detecta-los fallos.
- Informe-memoria da realización da montaxe. Elaboración, estruturando os resultados obtidos para unha adecuada documentación.

Selección e aplicación de probas de funcionalidade nos prototipos realizados

- Técnicas de análise estático e funcional. Identificación das características de cada un.
- Probas funcionais e de axuste.
 - Selección da alimentación eléctrica correcta.
 - Realización de conexións cos aparellos de verificación, aplicando as probas establecidas na documentación técnica.
 - Realización de axustes e posta a punto.
- Informe-memoria das probas funcionais e axustes. Elaboración, estruturando os resultados obtidos para unha adecuada documentación.

Realización de probas de fiabilidade nos prototipos realizados

- Características de calidade e fiabilidade. Análise dos conceptos básicos.
- Control de calidade nun prototipo electrónico.
 - Selección do procedemento adecuado.
 - Comprobación dos materiais de entrada.
 - Comprobación das diferentes fases da montaxe e inspección final.
 - Aplicación do procedemento de control de calidade establecido.
- Control de fiabilidade nun prototipo electrónico.
 - Selección do procedemento adecuado.
 - Cuantificación da vida do produto.
 - Realización dunha análise térmica, mecánica, eléctrica e de humidade.
 - Aplicación do procedemento de control de fiabilidade establecido.

- Informe-memoria das actividades realizadas. Elaboración, estruturando os resultados obtidos para unha adecuada documentación.

1.3.2 Contidos conceptuais

DESENVOLVEMENTO DE PROXECTOS DE PRODUTOS ELECTRÓNICOS

Fases no deseño e desenvolvemento dun proxecto electrónico:

- Especificacións do proxecto. Reunión de datos.
- Normativa para seguir segundo cada caso.
- Documentación técnica e/ou bases de datos.
- Tecnoloxías para utilizar.
- Simbología normalizada.
- Técnicas de estimación de tempos e custos.
- Modelos de soportes, esquemas e listaxes.
- Programas de simulación electrónica, instrucións e funcionamento.
- Ferramentas e instrumentos necesarios nas técnicas de montaxe rápida para a construción de maquetas electrónicas.
- Programas para sistemas microcontrolados.
- Equipos de desenvolvemento.

Posta a punto e documentación:

- Especificacións de calidade axustadas á normativa.
- Instrumentos, equipos e programas para as probas e posta a punto.
- Tipos de formatos e soportes para a elaboración da documentación técnica:
 - Lista de materiais.
 - Esquemas, planos, instrucións de montaxe e posta a punto.
 - Probas funcionais, de calidade e de fiabilidade.
 - Memoria descritiva e programas.

Planificación e xestión de proxectos:

- Técnicas de desenvolvemento de proxectos.

Aspectos organizativos.

- Métodos para o estudo da viabilidade.
- Técnicas de xestión de recursos, planificación de tempos e estimación de custos. Control de compras e materiais.
- Técnicas utilizadas na planificación e seguimento de proxectos.
- Informes e documentación na finalización e entrega de proxectos.

Técnicas de deseño de circuítos impresos

- Programas informáticos usados na edición de esquemas electrónicos.
- Tipoloxía e características das placas de circuíto impreso.
- Compoñentes das librerías. Deseño e creación, se non existen.
- Técnicas de interconexión dos diferentes compoñentes.
- Diferentes encapsulados na superficie da placa de circuíto impreso definida.
- Técnicas de trazado manual e/ou automático de pistas.
- Arquivos necesarios para a fabricación de tarxetas.

Construción de placas de circuíto impreso

- Programas informáticos usados na configuración de circuítos impresos.
- Características funcionais da placa requiridas polo deseño.
- Diferentes axentes de revelado, gravado, insolado e decapado.
- Documentación técnica precisa para a realización da placa.
- Maquinas, ferramentas e materiais necesarios.

Montaxe de compoñentes nas placas de circuíto impreso

- Compoñentes empregados na montaxe.
- Técnicas de montaxe de placas.
- Técnicas de soldadura.
- Técnicas de inspección visual e de limpeza.

Probas funcionais e axustes

- Equipos de medida.
- Probas e axustes establecidos na documentación técnica da propia tarxeta.

Control de calidade e fiabilidade

- Plans de control de calidade e normas en vigor.
- Técnicas de análise de calidade, mecánicas e eléctricas.
- Ensaio de longa duración.
- Ensaio de curta duración.
- Efectos de temperatura nos compoñentes electrónicos.
- Métodos de cálculo e avaliación aproximada da vida dun prototipo.

1.3.3 Contidos actitudinais

DESENVOLVEMENTO E CONSTRUCCIÓN DE PROTOTIPOS ELECTRÓNICOS DESENVOLVEMENTO DE PROXECTOS DE PRODUTOS ELECTRÓNICOS

Traballo en grupo

- Colaboración entre os compoñentes do grupo, polo seu propio autocontrol.
- Valoración de calquera idea con cordialidade, respecto e tolerancia co resto dos compañeiros.
- Preocupación por axuda-los compañeiros en tarefas de aprendizaxe.
- Cooperación entre os compoñentes do grupo, segundo a tarefa para realizar e os medios dispoñibles.

Comportamento interpersonal

- Actitude de apertura cos compañeiros, intercambiando experiencias e procurando as mellores solucións para resolve-los traballos a realizar.
- Valoración de calquera idea con cordialidade, respecto e tolerancia co resto dos compañeiros.
- Rigor en argumentar as ideas propias para a resolución de cada traballo, contrastando cos compañeiros do grupo.

Optimización do traballo na realización de tarefas

- Preocupación pola autoorganización das secuencias nas operacións de desenvolvemento e construción de prototipos, tomando decisións de prioridade de traballo, buscando optimizala relación entre calidade e tempo.
- Espírito crítico na autoavaliación dos métodos de traballo empregados, buscando melloralos factores, calidade e tempo empregado.

Realización estruturada do traballo na execución de tarefas

- Preocupación pola orde no posto de traballo, dispoñendo as ferramentas, útiles e instrumentos no mellor lugar para ser empregados.
- Comportamento activo para conseguir acabados pulcros dos traballos, realizando a verificación visual e sistemática do produto final.
- Preocupación pola aplicación de procedementos normalizados na realización de tarefas de desenvolvemento e construción de prototipos.

Compromiso coas abrigas asociadas ó traballo

- Responsabilidade na conservación das máquinas, ferramentas, útiles e instrumentación, facendo por iniciativa propia o mantemento máis usual.
- Hábito de realizar as operacións de desenvolvemento e construción de prototipos seguindo normas e recomendacións de seguridade laboral e facendo un uso correcto das ferramentas e da instrumentación.

Avaliación de resultados

- Valoración dos traballos efectuados en función dos resultados, tempo utilizado, metodoloxía, os aparellos e a instrumentación utilizada nas tarefas para realizar.

- Responsabilidade para emitir un xuízo sobre os criterios utilizados á hora de finalizar un traballo encomendado.
- Preocupación por utilizar a autoavaliación como ferramenta para a mellora das capacidades persoais.

Coordinación dos recursos humanos

- Compromiso cos aplicación da lexislación laboral e as condicións dos contratos de cada traballador.
- Preocupación por facilitar a relación entre traballadores, intercambiando coñecementos e actitudes positivas.
- Actitude de apertura das propostas de mellora da calidade e da produtividade do servizo por parte dos traballadores.

1.3.4 Dominio profesional

DESENVOLVEMENTO E CONSTRUCCIÓN DE PROTOTIPOS ELECTRÓNICOS

DESENVOLVEMENTO DE PROXECTOS DE PRODUTOS ELECTRÓNICOS

Medios de produción ou tratamento da información

- Material de debuxo. Calculadora. Ordenador. Periféricos de ordenador (impresora, trazador gráfico, táboa dixitalizadora). Programas informáticos de debuxo e deseño asistidos por ordenador (CAD-CAE) para o deseño e simulación de placas de CI. Arquivadores de planos. Material de oficina xeral. Ferramentas manuais para traballos eléctricos e mecánicos (alicates, desaparafusadores, pelacables, soldador, trade). Estación de soldadura e desoldadura de compoñentes electrónicos (de inserción e de montaxe superficial). Fototrazador gráfico. Pequenas máquinas para metalizado de trades nos CI. Pantallas serigráficas para CI. Pequenas máquinas para montaxe de compoñentes en CI para prototipos (manuais, automáticas). Pequena máquina de control numérico (CNC) para trades. Pequeno forno de refusión. Instrumentos de medida e verificación electrónica (polímetro, osciloscopio, frecuencímetro, xeradores de BF e AF, fontes de alimentación, analizador de espectros, inxector e sonda lóxica, analizador de estados lóxicos). Instrumentación para rexistro de parámetros. Instrumentación para ensaios de fiabilidade.

Materiais e produtos intermedios

- Esquemas electrónicos e listas de materiais. Condutores eléctricos e elementos de interconexión. Compoñentes electrónicos. Ficheiros para a realización do fotogravado de CI. Materiais para o metalizado, fotogravado e atacado químico de placas de CI. Materiais para a soldadura de compoñentes en placas de CI. Prototipos de circuítos electrónicos. Follas de medida e informes de probas. "Software" de proba.

Principais resultados do traballo (produtos e/ou servizos)

- Prototipos de circuítos electrónicos analóxicos e dixitais. Documentación correspondente ó deseño físico do produto electrónico (esquemas electrónicos, planos do deseño físico dos CI - disposición de compoñentes e serigrafía, pistas das distintas capas, máscaras de soldadura, plano de trades-, ficheiros para fototrazador e máquina de trade CNC). Informes de probas funcionais e de fiabilidade.

Procesos, métodos e procedementos

- Procedementos de disposición de compoñentes en e de trazado de pistas en placas de CI (manual e automático). Procedementos de obtención de documentación para elaboración de placas de CI (en soportes papel e informáticos). Procesos de fabricación de CI (metalizado de trades, preparación de pantallas serigráficas e protección de CI). Procedementos de programación de CNC para tradeado de placas de CI. Métodos de soldadura de compoñentes de inserción e de montaxe superficial (SMD). Proceso de montaxe electrónica manual (dobrado, inserción, corte de terminais, soldadura e desoldadura). Proceso automático de montaxe de compoñentes electrónicos (programación de máquinas de posicionamento de compoñentes). Procedementos de gravado de dispositivos electrónicos programables (memorias, dispositivos lóxicos programables -PLD-, microcontroladores). Procedementos de medida de magnitudes electrónicas no dominio do tempo e da frecuencia. Procedementos de axuste de circuítos electrónicos. Procedementos xerais de documentación.

Información (natureza, tipo e soportes)

- Especificacións técnicas do produto. Normativa técnica e de calidade aplicable ó produto. Manuais de circuítos electrónicos. Normativa interna de utilización de compoñentes electrónicos. Manuais internos de montaxe e interconexión. Manuais internos sobre procedementos de axuste. Manuais internos sobre procedementos de análises de fiabilidade. Manuais internos sobre tipo e contido de documentación de produtos electrónicos (en soporte de papel e informático).

Persoal e/ou organizacións destinatarias

- Departamento e/ou persoal de deseño. Departamento e/ou persoal de industrialización.
- Departamento e/ou persoal de calidade. Cliente. Organismos oficiais de homologación.

1.4 Aspectos metodolóxicos

- Tipo da actividade presencial, de aprendizaxe, práctica, escrita e individual e de grupo.
- Recursos:
 - Materiais: plataformas de programación en linguaxe C simuladores e programadores de PIC, aplicación informática OrCAD, taller de fabricación de circuítos impresos, estación de montaxe e soldadura, instrumentación electrónica de medicións e probas.
 - Espazos: taller de electrónica do centro educativo nas horas presenciais.
- Contorno de comunicación: Aulas dos IES.

1.4.1 Requisitos mínimos de espazos e instalacións

De conformidade co establecido no R.D. 777/1998, do 30 de abril, o ciclo formativo de formación profesional de grao superior de desenvolvemento de produtos electrónicos require, para a impartición das ensinanzas relacionadas neste decreto, os seguintes espazos mínimos:

Espazo formativo	Superficie(30alumnos)	Superficie(20alumnos)	Grao utilización
Aula Técnica	90m ²	60 m ²	20%
Taller Electricidade-Electrónica	120 m ²	90 m ²	65%
Aula Polivalente	60 m ²	40 m ²	15%

A superficie indicada na segunda columna da táboa corresponde ó número de postos escolares establecido no artigo 35 do R.D. 1004/1991, do 14 de xuño. Poderán autorizarse unidades para menos de trinta postos escolares, polo que será posible reducilos espazos formativos proporcionalmente ó número de alumnos, tomando como referencia para a determinación das superficies necesarias as cifras indicadas nas columnas segunda e terceira da táboa.

O grao de utilización expresa en tanto por cento a ocupación en horas do espazo prevista para a impartición das ensinanzas, por un grupo de alumnos, respecto da duración total destas ensinanzas. Na marxe permitida polo grao de utilización, os espazos formativos establecidos poden ser ocupados por outros grupos de alumnos que cursen o mesmo ou outros ciclos formativos, ou outras etapas educativas.

En todo caso, as actividades de aprendizaxe asociadas ós espazos formativos (coa ocupación expresada polo grao de utilización) poderán realizarse en superficies utilizadas tamén para outras actividades formativas afíns. Non debe interpretarse que os diversos espazos formativos identificados deban diferenciarse necesariamente mediante cerramentos.

2. Programación de Pic's en Linguaxe C baixo protocolos SPI e I2C

2.1 Introducción

Nesta actividade describiranse os diferentes procedementos e estratexias para a programación de Pic's baixo protocolos SPI e I²C mediante un programador comercial, para seguir a continuación implementando un entrenador de programación de uso didáctico para Pic's no que se desenvolverán os diferentes programas que serán desenvolvidos nas aulas co alumnado.

Nesta **actividade** distínguense tres partes:

Primeira: Descrición do traballo feito co relator e o desenvolvemento dos programas en C para a súa utilización no programador comercial.

Segunda: Construción do programador mediante os PCB's correspondentes co ORCAD,. Isto levarase logo á aula no módulo de desenvolvemento e construción de prototipos electrónicos..

2.2 Programas Feitos co Relator en Linguaxe C baixo protocolos I2C e SPI

O comezo deste grupo de traballo dedicamos as primeiras reunións a poñernos ao día da programación en linguaxe C de Pic's baixo protocolos de comunicación I2C e SPI, contando coas horas de docencia en que traballamos co noso relator, Fernando Rodríguez Rodríguez, aprendemos os rudimentos deste tipo de programas e centramos os nosos esforzos nas características destes protocolos e finalmente no estudo do programador empregado para as comprobación dos programas.

No estudo destes protocolos podemos facer referencia a algunha documentación que pode ser de utilidade para todos aqueles que queiran preparar e desenvolver estas prácticas.

Con respecto ao manual e documentación do programado, utilizamos o:

Development Kit For the PICMCU Exercise Book Embedded Serial Busses.

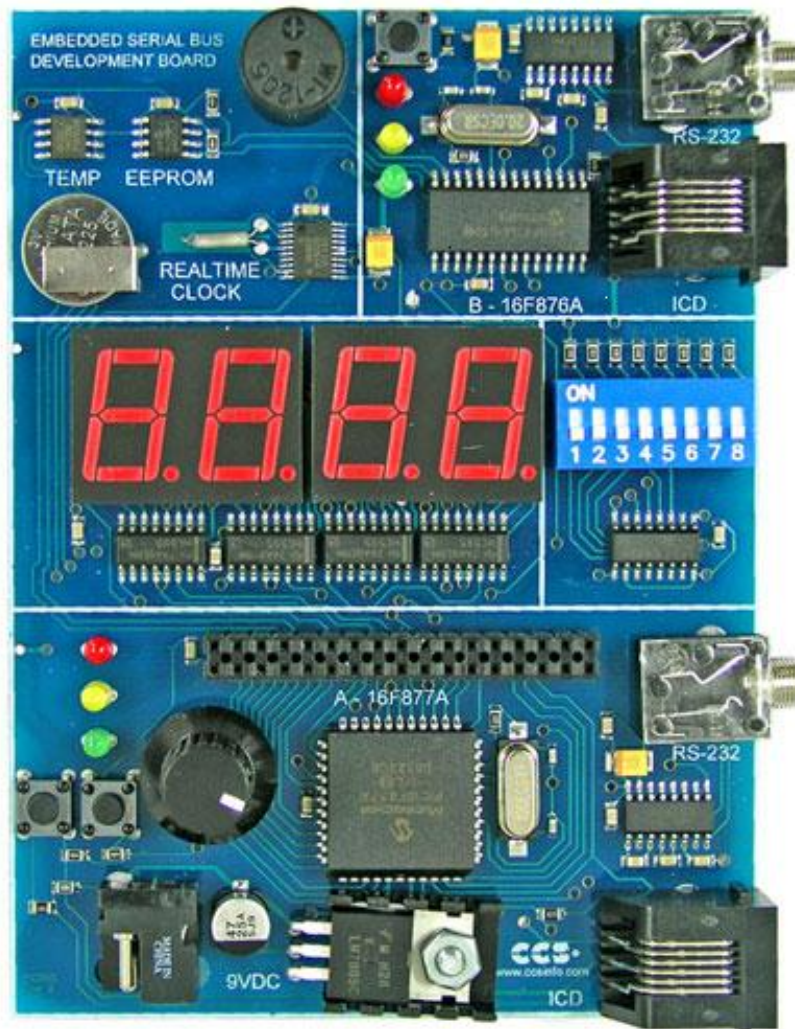
E algunhas das páxinas web mais utilizadas para esta actividade:

<http://www.pic-c.com>

<http://www.piclist.com>

<http://www.ccsinfo.com/download>

O programador que mostramos a continuación foi o “Embedded Serial Busses” baixo a plataforma PIC-C co compilador CCS.



O resultado destes traballos van ser descritos a continuación cos programas correspondentes a cada un, para a visualización dos mesmos anexamos unha carpeta con vídeos que se adxunta na documentación en soporte informático.

2.2.1 PARPADEO DE LED VERDE POR SEGUNDO.

```
/******  
* FICHEIRO: EscintilaLEDverde.c  
* CONTIDO: Programa de proba da placa, sobre o PIC16F887A, fai escintilar  
* o LED verde (etiqueta A5(pin 8)) cada segundo.  
* FUNCIONES:  
* main(): Bucle infinito que apaga e acende o LED verde cada segundo.  
*****/  
/***** CONFIGURACIÓN ESPECÍFICA DA PLACA OU DO EXERCICIO:  
* Frecuencia de oscilación do PIC: clock=10000000  
* Pin do cátodo do LED verde: LED_VERDE  
*****/  
  
// Contorno básico do hardware:  
#include <16f877A.h> // Usamos PIC16F877A  
#device ICD=TRUE // Usamos depurador ICD  
#fuses HS,NOLVP,NOWDT,PUT // Non aceptamos programación nivel baixo  
#use delay (clock=10000000) // Frecuencia de oscilación do PIC: 10Mhz  
// Pins dos cátodos dos LEDs  
#define LED_VERDE PIN_A5 // Conectado no pin 8 (A5) do PIC  
//----- Declaración de funcións: -----  
  
void main();  
//-----  
/*****  
* FUNCIÓN: main()  
* Bucle infinito que leva sucesivamente a saída PIN_A5(8) a niveis  
* baixo=acendido, alto=apagado. Espera 1 segundo entre cambios.  
*****/  
  
void main ()  
{  
while (TRUE)  
{  
output_low (LED_VERDE); // Acende o LED verde  
delay_ms (1000); // Espera 1 segundo  
output_high (LED_VERDE); // Apaga o LED verde  
delay_ms (1000); // Espera 1 segundo  
}}  
}
```

2.2.2 PARPADEO DE TRES LEDS SECUENCIAL VARIANDO O TEMPO

```
/******  
* FICHEIRO: EscintilaLEDsecuencial.c  
* CONTIDO: Programa de proba da placa, sobre o PIC16F887A, acende os LEDs  
* vermello, amarelo e verde consecutivamente, variando o período e o ciclo.  
* Comeza ao pulsar o interruptor A4 (etiquetado no manual como A0 pero  
* conectado ao pin A4 do PIC)e remata ao pulsar o interruptor E0 do PIC .  
* FUNCÍONS:  
* void main(): Bucle á espera de pulsar A4. Bucle de activación de LEDs ata  
* pulsado do E0.  
* void EscintilaLED(int8 led, int16 período, int8 ciclo): Escintila o LED  
* segundo a combinación do período (milisegundos) e do ciclo de traballo(%).  
*****/  
  
/***** CONFIGURACIÓN ESPECÍFICA DA PLACA OU DO EXERCICIO:  
* Frecuencia de oscilación do PIC: clock=10000000  
* Pins dos cátodos dos LEDs: LED_VERDE, LED_AMARELO, LED_BERMELLO  
*****/  
  
#include <16f877A.h> // Usamos PIC16F877A  
#device ICD=TRUE // Usamos depurador ICD  
#fuses HS,NOLVP,NOWDT,PUT // Non aceptamos programación nivel baixo  
#use delay (clock=10000000) // Frecuencia de oscilación do PIC: 10Mhz  
// Pins dos cátodos dos LEDs:  
#define LED_BERMELLO PIN_B1 // Conectado no pin 37 (B1) do PIC  
#define LED_AMARELO PIN_B4 // Conectado no pin 41 (B4) do PIC  
#define LED_VERDE PIN_A5 // Conectado no pin 8 (A5) do PIC  
//----- Declaración de funcións: -----  
  
void main();  
void EscintilaLED(int8 led, int16 periodo, int8 ciclo);  
/******  
* FUNCIÓN: void main()  
* Arranque: Bucle de espera de cambio a alto no pulsador A4 (pin 7).  
* Bucle do-while (ao menos unha vez): Activa un a un os LEDs vermello, amarelo  
* e verde variando o período e o ciclo de traballo na seguinte secuencia:  
* - Acende: 0.5s, apaga: 0.5s (periodo 1000ms, ciclo 50%)  
* - Acende: 0.5s, apaga: 1.5s (periodo 2000ms, ciclo 25%)  
* - Acende: 1s, apaga: 1s (periodo 2000ms, ciclo 50%)  
* - Acende: 1.5s, apaga: 0.5s (periodo 2000ms, ciclo 75%)  
*****
```

```

*   - Acende: 0.25s, apaga: 0.25s (periodo 500ms, ciclo 50%)
*   Condición de saída do bucle do-while: cambio a alto no pulsador E0 (pin 9)
*****/

void main()
{
  while (input(PIN_A4)) // Espera cambio a alto en A4.
  // Escintila os LEDs variando o ciclo de traballo
  do {
    // Acende: 0.5s, apaga: 0.5s (periodo 1000ms, ciclo 50%)
    EscintilaLED(LED_BERMELLO, 1000, 50);
    EscintilaLED(LED_AMARELO, 1000, 50);
    EscintilaLED(LED_VERDE, 1000, 50);
    // Acende: 0.5s, apaga: 1.5s (periodo 2000ms, ciclo 25%)
    EscintilaLED(LED_BERMELLO, 2000, 25);
    EscintilaLED(LED_AMARELO, 2000, 25);
    EscintilaLED(LED_VERDE, 2000, 25);
    // Acende: 1s, apaga: 1s (periodo 2000ms, ciclo 50%)
    EscintilaLED(LED_BERMELLO, 2000, 50);
    EscintilaLED(LED_AMARELO, 2000, 50);
    EscintilaLED(LED_VERDE, 2000, 50);
    // Acende: 1.5s, apaga: 0.5s (periodo 2000ms, ciclo 75%)
    EscintilaLED(LED_BERMELLO, 2000, 75);
    EscintilaLED(LED_AMARELO, 2000, 75);
    EscintilaLED(LED_VERDE, 2000, 75);
    // Acende: 0.25s, apaga: 0.25s (periodo 500ms, ciclo 50%)
    EscintilaLED(LED_BERMELLO, 500, 50);
    EscintilaLED(LED_AMARELO, 500, 50);
    EscintilaLED(LED_VERDE, 500, 50);
  } while (input(PIN_E0)); // Repite ata que E0 cambia a alto.
}

*****/

*   FUNCIÓN: void EscintilaLED(int8 led, int16 período, int8 ciclo)
*   Escintila o LED segundo a combinación do período (milisegundos) e do ciclo
*   de traballo (%) segundo as ecuacións:.
*   - acendido = período * ciclo/100 (milisegundos)
*   - apagado = período - acendido (milisegundos)
*****/

void EscintilaLED(int8 led, int16 período, int8 ciclo)

```

```

{
    int16 acendido; // Tempo acendido en milisegundos
    int16 apagado; // Tempo apagado en milisegundos
    // Cálculo dos tempos
    acendido = período * (ciclo/100);
    apagado = período - acendido;
    output_low (led); // Acende o LED levando a baixo a saída
    delay_ms (acendido); // Espera o tempo de manter acendido
    output_high (led); // Apaga o LED levando a alto a saída
    delay_ms (apagado); // Espera o tempo de manter apagado
}

```

2.2.3 RELOXO CONTADOR DE SEGUNDOS EN FORMATO SSSS

```

/*****
* FICHEIRO: DisplaySegundos.c
* CONTIDO: Programa sobre o PIC16F877A que mostra un contador de segundos
* (de 0000 a 9999) nos 4 displays 7-segmentos.
* FUNCIONES:
* void main(): Contador cíclico de 0 a 9999 mostrado en los 4 displays.
* void AmosaEnDisplays(int16 numero): Recibe un número entre 0 e 9999 para
* amosalo nos 4 displays da placa usando a configuración de rexistros de
* desprazamento (74595).
*****/

/*****
* Fundamento: Cada un dos 4 displays-7segmentos (cátodo común) ten asociado
* un CI 74HC595 presentando nas súas 8 saídas os valores dos 7 segmentos e
* do punto (nivel alto = acendido). O CI 74595 é un rexistro de desprazamento
* de 8 bits; os datos entran en serie e poden saír en serie ou en paralelo.
* A carga dos "díxitos" realízase entrando en serie e pasando en serie do
* rexistro do díxito máis significativo cara ao seguinte rexistro, sen afectar
* as saídas. En 8x4 ciclos de reloxo temos os 4 díxitos cargados nos rexistros.
* A saída definitiva del díxito cara o seu display é en paralelo e realízase
* cargando no LATCH o dato xa introducido rexistro e habilitando a súa saída.
*****/

/*****
* Implementación: As funcións específicas desta configuración de rexistros
* as temos no include (../Drivers/74595.c). Temos que definir os pines do
* microprocesador correspondentes ás entradas dos rexistros e o número de

```

```

* rexistros; estes defines serán usados despois nas funcións do driver:
* - entrada serie de datos (pin SER do rexistro do display de máis peso:
* #define EXP_OUT_DO ) PIN_D1
* - entrada de reloxo para introducir os datos (pin SCLK de tódolos rexistros):
* #define EXP_OUT_CLOCK PIN_D0
* - entrada de reloxo para pasar os datos xa introducidos no rexistros ao LATCH
* (flanco de subida no pin RCLK dos rexistros) e para habilitar a saída
* (nivel LOW na entrada *OE de tódolos rexistros):
* #define EXP_OUT_ENABLE PIN_D2
* - número de rexistros:
* #define NUMBER_OF_74595 4
* Usamos a función encargada da carga serie dos datos, que recibe os díxitos
* codificados nun vector de 4 bytes: write_expanded_outputs(*int8 eo).
* A función realiza os seguintes pasos:
* - Deshabilita as saídas dos rexistros (é triestado) pasando a LOW ..ENABLE.
* - Prepara o reloxo para poder dar a súa primeira clock (flanco ascendente)
* pasando a LOW ..CLOCK.
* - Nun bucle de 8bits x 4rexistros, pon o valor de bit na saída ..DO e activa
* o seu ingreso nos rexistros cun clock pasando ..CLOCK a HIGH e volvéndoo a
* LOW para preparar o seguinte bit.
* - Pasa os datos xa introducidos nos rexistros aos LATCHs pasando ..ENABLE a
* HIGH (neste caso afecta ao pin RCLK dos rexistros) e activa as saídas
* pasando a LOW ..ENABLE (neste caso afecta ao pin *OE dos rexistros)
*****/
/*****

* Codificación dos díxitos cara ao Display-7segmentos:
* O MSB (bit máis significativo), que entra o primeiro en serie queda aloxado
* na súa posición final (saída QH) despois de 8 ciclos de reloxo.
* Os bits, de MSB a LSB son:
* DP.G.F.E | D.C.B.A e expresarémoslos en hexadecimal. Exemplos:
* 0.0.1.1 | 1.1.1.1 = 0x3F == 0
* 0.0.0.0 | 0.1.1.0 = 0x06 == 1
* 0.1.0.1 | 1.0.1.1 = 0x5B == 2
* 1.1.1.1 | 0.0.1.0 = 0x4F == 3
* Os valores dos 10 díxitos os codificamos nun vector de 10 bytes:
* - const byte dixitos[10] = {0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F}
*****/
/***** CONFIGURACIÓN ESPECÍFICA DA PLACA OU DO EXERCICIO:

```

```

* Frecuencia de oscilación do PIC: clock=10000000
* Adapta a velocidade do reloxo: VELOCIDADE
* Pins e número dos rexistros de desprazamento (CI 74595): EXP_OUT_DO,
* EXP_OUT_CLOCK, EXP_OUT_ENABLE, NUMBER_OF_74595
*****/

// Contorno básico do hardware
#include <16f877A.h> // Usamos PIC16F877A
#define ICD=TRUE // Usamos depurador ICD
#define HS,NOLVP,NOWDT,PUT // Non aceptamos programación nivel baixo
#define use delay (clock=10000000) // Frecuencia de oscilación do PIC: 10Mhz
// Cambiar 1000 (conta segundos) a 100 para que corra 10 veces máis
#define VELOCIDADE 1000
// Defines e include do driver da configuración de rexistros 74595
#define EXP_OUT_DO PIN_D1 // SER (Dato de entrada serie)
#define EXP_OUT_CLOCK PIN_D0 // SCLK (Reloxo, flanco ascendente. Entra dato)
#define EXP_OUT_ENABLE PIN_D2 // RCLK (Reloxo, flanco ascendente. Dato ao LATCH)
// e *OE (LOW: Habilita a saída do dato paralelo)
#define NUMBER_OF_74595 4 // Temos 4 displays e, polo tanto, 4 rexistros
#include <74595.c> // Include do ../Driver/74595.c
// Codificación dos díxitos decimais do '0' ao '9' cara a activar os displays
const int8 dixitos[10] = {0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};
//----- Declaración de funcións: -----

void main();
void AmosaEnDisplays(int16 numero);
/*****
* FUNCIÓN: void main()
* Bucle infinito que incrementa un contador cíclico de 0 a 9999 cada segundo
* e mostra o seu valor en los 4 displays 7-segmentos de la placa.
* NOTA: A función non é un contador preciso xa que non ten en conta o tempo
* de execucións das instrucións (4xfosc)
*****/

void main()
{
    // Contador de segundos
    int16 segundos = 0;
    output_low(PIN_D6); //disable RTC
    // Bucle infinito

```

```

while(true)
{
    // Amosa nos 4 Displays os segundos, de 0000 a 9999
    AmosaEnDisplays(segundos);
    // Si chegamos aos 9999 segundos, reiniciamos o contador
    if (++segundos == 10000)
        segundos = 0;
    // Espera de 1 segundo = 1000 milisegundo (ou 0.1s segundo o #define)
    delay_ms(VELOCIDADE);
}
}

/*****
* FUNCIÓN: void AmosaEnDisplays(int16 numero)
* Recibe un número entre 0 e 9999 para amosalo nos 4 displays da placa.
* - Garda nun vector cada díxito decimal.
* - Escribe nos displays os díxitos usando os rexistros de desprazamento.
* Cada díxito decimal é codificado ao valor requirido para amosalo no display
* A función write_expanded_outputs(*int eo) está defina no include do driver 74595.c.
* - Nos displays amosa do 0000 ao 9999
*****/

void AmosaEnDisplays(int16 numero)
{
    int8 amosa_dixitos[4];
    // Calcula cada díxito decimal e obtén o seu valor codificado vector díxitos[]
    // Copia o resultado no vector amosa_dixitos[] que pasará á función do
    // driver 74595.c para mostralos nos displays.
    amosa_dixitos[0] = dixitos[numero / 1000 % 10];
    amosa_dixitos[1] = dixitos[numero / 100 % 10];
    amosa_dixitos[2] = dixitos[numero / 10 % 10];
    amosa_dixitos[3] = dixitos[numero % 10];
    write_expanded_outputs(amosa_dixitos);
}

```

2.2.4 RELOXO DE MINUTOS E SEGUNDOS FORMATO MM SS

```
/******  
* FICHEIRO: DisplayReloxo.c  
* CONTIDO: Programa sobre o PIC16F877A que mostra un reloxo de segundos e  
* minutos nos 4 displays 7-segmentos, co formato mm ss (de 00 00 a 99 99,  
* reiniciando despois de 59 cada grupo).  
* FUNCÍONS:  
* void main(): Contador cíclico de de 00 00 a 99 99 mostrado en los 4 displays.  
* void AmosaEnDisplays(int16 numero): Recibe un número entre 0 e 9999 para  
* amosalo nos 4 displays da placa usando a configuración de rexistros de  
* desprazamento do driver: ../Driver/74595.c.  
* NOTA: Para ver o funcionamento completo e non ter que esperar 1 hora, pódese  
* cambiar o define VELOCIDADE de 1000 a 100 para que corra 10 veces máis.  
*****/  
  
/***** CONFIGURACIÓN ESPECÍFICA DA PLACA OU DO EXERCICIO:  
* Frecuencia de oscilación do PIC: clock=10000000  
* Adapta a velocidade do reloxo: VELOCIDADE  
* Pins e número dos rexistros de desprazamento (CI 74595): EXP_OUT_DO,  
* EXP_OUT_CLOCK, EXP_OUT_ENABLE, NUMBER_OF_74595  
*****/  
  
// Contorno básico do hardware  
#include <16f877A.h> // Usamos PIC16F877A  
#device ICD=TRUE // Usamos depurador ICD  
#fuses HS,NOLVP,NOWDT,PUT // Non aceptamos programación nivel baixo  
#use delay (clock=10000000) // Frecuencia de oscilación do PIC: 10Mhz  
// Cambiar 1000 (conta segundos) a 100 para que corra 10 veces máis  
#define VELOCIDADE 1000  
  
/******  
* Fundamento: Amosa díxitos nos 4 displays mediante o driver: ../Drivers/74595.c  
*****/  
  
// Defines e include do driver da configuración de rexistros 74595  
#define EXP_OUT_DO PIN_D1 // SER (Dato de entrada serie)  
#define EXP_OUT_CLOCK PIN_D0 // SCLK (Reloxo, flanco ascendente. Entra dato)  
#define EXP_OUT_ENABLE PIN_D2 // RCLK (Reloxo, flanco ascendente. Dato ao LATCH)  
  
// e *OE (LOW: Habilita a saída do dato paralelo)  
#define NUMBER_OF_74595 4 // Temos 4 displays e, polo tanto, 4 rexistros  
#include <74595.c> // Include do ../Driver/74595.c
```

```
// Codificación dos díxitos decimais do '0' ao '9' cara a activar os displays
const int8 dixitos[10] = {0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};

//----- Declaración de funcións: -----

void main();

void AmosaReloxoEnDisplays(int8 minutos, int8 segundos);

//-----

/*****
* FUNCIÓN: void main()
* Bucle infinito mostraremos nos 4 displays un reloxo de formato ss mm,
* ciclicamente desde 00 00 ata 59 59 (reinicio cada hora).
* Composto dun bucle de contar segundos de 00 a 56, incrementando os segundos
* de 00 a 59.
* NOTA: Para facilitar as probas do programa, podemos cambiar o define
* VELOCIDADE de 1000 (segundos) a 100 (10 veces máis rápido)
* NOTA: A función non é un contador preciso xa que non ten en conta o tempo
* de execucións das instrucións (4xfosc)
*****/

void main()
{
    // Contadores de segundos e minutos. Amosarase nos displays co formato: mm ss
    int8 segundos = 0;
    int8 minutos = 0;
    output_low(PIN_D6); //disable RTC (RealTimeClock)
    // Bucle infinito do reloxo
    while (true) {
        // Reinicio do contador de minutos cada 60 minutos (de 00 a 59 en mm)
        minutos = 0;
        while(minutos < 60) {
            // Reinicio do contador de segundos cada 60 segundos (de 00 a 59 en ss)
            segundos = 0;
            while (segundos < 60) {
                // Amosamos minutos e segundo nos displays (mm ss)
                AmosaReloxoEnDisplays(minutos, segundos);
                // Esperamos os milisegundos indicados no define VELOCIDADE
                // 1000 é o normal para contar segundos
                // 100 é dez veces máis rápido para comprobar o programa.
                delay_ms(VELOCIDADE);
            }
        }
    }
}
```

```

    segundos++;    // Incrementamos o contador de segundos
}
minutos++;    // Incrementamos o contador de minutos
} } }

/*****
* FUNCIÓN: void AmosaReloxoEnDisplays(int8 minutos, int8 segundos)
* Recibe minutos (ente 00 e 59) y segundos (entre 00 e 59) para amosalos nos
* 4 displays da placa, no formato: mm ss.
* - Garda nun vector cada díxito decimal, minutos e segundos.
* - Escribe nos displays os díxitos usando os rexistros de desprazamento.
* Cada díxito decimal é codificado ao valor requirido para amosalo no display
* A función write_expanded_outputs(*int eo) está defina no include do
* driver 74595.c.
* - Nos displays amosa do 00 00 ao 59 59.
*****/

void AmosaReloxoEnDisplays(int8 minutos, int8 segundos)
{
    int8 amosa_dixitos[4];
    // Calcula cada díxito decimal e obtén o seu valor codificado vector dixitos[]
    // Copia o resultado no vector amosa_dixitos[] que pasará á función do
    // driver 74595.c para mostralos nos displays.
    amosa_dixitos[0] = dixitos[minutos / 10 % 10];
    amosa_dixitos[1] = dixitos[minutos % 10];
    amosa_dixitos[2] = dixitos[segundos / 10 % 10];
    amosa_dixitos[3] = dixitos[segundos % 10];
    write_expanded_outputs(amosa_dixitos);
}

```

2.2.5 RELOXO PROGRAMABLE POR BOTONS EN FORMATO MM SS

```
/******  
* FICHEIRO: ReloxoProgramable.c  
* CONTIDO: Programa sobre o PIC16F877A que mostra un reloxo de segundos-minutos  
* sobre os 4 displays 7-segmentos, con formato de saída: mm ss, e que se pode  
* programar o seu valor inicial con dous botóns, só ao inicio do programa.  
* BOTON1 - PIN_E0: Premer 250ms(T_PREM_B1), cambia modos programación.  
* BOTON2 - PIN_A4: Premer 100ms(T_PREM_B2), incrementa segundos ou minutos.  
* FUNCIONES:  
* void main(): Indica nos displays o arrinque, a opción de programación e  
* activa o reloxo.  
* - Indica o arrinque do programa escintilando -- -- nos displays. (2s)  
* - Indica a opción de programación escintilando 00 00 nos displays. (max.4s)  
* - Pasa o control á función ModoProgramacion().  
* - Comeza a correr o reloxo a partir dos minutos e segundos programados.  
* void EscintilaReloxoDisplays(int8 minutos, int8 segundos, int16 período,  
* int8 ciclo): Acende os 4 displays co valor mm ss (entre 00 00 e 59 59)  
* durante período*ciclo/100 e os apaga o resto do período.  
* int8 ModoProgramacion(int8 *minutos_ini, int8 *segundos_ini): Permite  
* programar os valores iniciais do segundos e minutos cos botóns:  
* BOTON1 (cambios de modo de programación), BOTON2 (incremento dos  
* segundos ou minutos iniciais.  
* void ReloxoProgramado(int8 minutos, int8 segundos): Reloxo que parte dos  
* valores iniciais dados polos argumentos.  
* void AmosaReloxoEnDisplays(int8 minutos, int8 segundos): Amosa mm ss  
* nos 4 displays usando a configuración de rexistros de desprazamento (74595).  
* void ApagaDisplays(): Apaga os displays deshabilitando os rexistros 74595.  
* int8 Pulsada(int8 entrada, int8 tempo): Determina unha pulsación en entrada  
* evitando o efecto rebote ou aumentando o control medindo de novo tras tempo  
*****/  
  
/***** CONFIGURACIÓN ESPECÍFICA DA PLACA OU DO EXERCICIO:  
* Frecuencia de oscilación do PIC: clock=10000000  
* Adapta a velocidade do reloxo: VELOCIDADE  
* Pins e número dos rexistros de desprazamento (CI 74595): EXP_OUT_DO,  
* EXP_OUT_CLOCK, EXP_OUT_ENABLE, NUMBER_OF_74595, GUION  
* Pins dos botóns: BOTON1, BOTON2  
* Tempo de verificación ou control de botón premido: T_PREM_B1, T_PREM_B2
```

```

*****/

// Contorno básico do hardware
#include <16f877A.h>          // Usamos PIC16F877A
#define ICD=TRUE             // Usamos depurador ICD
#define fuses HS,NOLVP,NOWDT,PUT // Non aceptamos programación nivel baixo
#define use delay (clock=10000000) // Frecuencia de oscilación do PIC: 10Mhz
// Cambiar 1000 (conta segundos) a 100 para que corra 10 veces máis
#define VELOCIDADE 1000

// Control da programación dos valores iniciais do reloxo
#define BOTON1 PIN_E0 // Cambios do modo de programación.
#define BOTON2 PIN_A4 // Incremento dos segundos ou dos minutos
#define ENTRADA_H 1 // Entrada con valor High (premido o botón)
#define T_PREM_B1 250 // Tempo (ms) para comprobar o botón (antirrebotes)
#define T_PREM_B2 100 // Tempo (ms) para comprobar o botón (antirrebotes)

/*****
* Fundamento: Amona díxitos nos 4 displays mediante o driver: ..Drivers/74595.c
*****/

// Defines e include do driver da configuración de rexistros 74595
#define EXP_OUT_DO PIN_D1 // SER (Dato de entrada serie)
#define EXP_OUT_CLOCK PIN_D0 // SCLK (Reloxo, flanco ascendente. Entra dato)
#define EXP_OUT_ENABLE PIN_D2 // RCLK (Reloxo, flanco ascendente. Dato ao LATCH)

// e *OE (LOW: Habilita a saída do dato paralelo)
#define NUMBER_OF_74595 4 // Temos 4 displays e, polo tanto, 4 rexistros
#define GUION 60 // Amosar nos LEDs guións (--)
#include <74595.c> // Include do ..Driver/74595.c

// Codificación dos díxitos decimais do '0' ao '9' cara a activar os displays
const int8 dixitos[10] = {0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};

//----- Declaración de funcións: -----

void main();

void EscintilaReloxoDisplays(int8 minutos, int8 segundos, int16 periodo,
int8 ciclo);

int8 ModoProgramacion(int8 *minutos_ini, int8 *segundos_ini);

void ReloxoProgramado(int8 minutos, int8 segundos);

void AmonaReloxoEnDisplays(int8 minutos, int8 segundos);

void ApagaDisplays();

int8 Pulsada(int8 entrada, int8 tempo);

//-----

```

```

/*****
* FUNCIÓN: void main()
* - Indica o arrinque do programa escintilando -- -- nos displays. (2s)
* - Indica a opción de programación escintilando 00 00 nos displays. (max.4s)
* - Pasa o control á función ModoProgramacion().
* - Comeza a correr o reloxo a partir dos minutos e segundos programados.
*****/

void main()
{
    int8 minutos = 0;
    int8 segundos = 0;
    int8 espera;

    // Indica o arrinque do programa escintilando -- -- nos displays.
    // Tempo: acendido=100ms, apagado=100ms, total=10*200=2000ms=2s
    espera = 10;
    while (espera-- > 0)
        EscintilaReloxoDisplays(GUION, GUION, 200, 50);
    // Amosa a opción de programación escintilando 00 00 nos displays.
    // Tempo: acendido=100ms, apagado=100ms, total_maximo=20*200=4s
    // Entra en modo programar si presión BOTON1 máis de T_PREM_B1 (Antirebotes)
    // Retorna os minutos e segundos iniciais nos argumentos da función.
    espera = 20;
    while (espera-- > 0)
    {
        EscintilaReloxoDisplays(0, 0, 200, 50);
        if (ModoProgramacion(&minutos, &segundos) == true)
            espera = 0;
    }
    // Bucle infinito do reloxo, comezando nos valores iniciais
    ReloxoProgramado(minutos, segundos);
}

/*****
* FUNCIÓN: EscintilaReloxoDisplays(int8 minutos, int8 segundos, int16 periodo,
int8 ciclo)
* Amosa nos 4 displays os 4 díxitos do dato (entre 0000 e 9999).
* Manteña os díxitos os milisegundos indicados: acendido = periodo * ciclo/100
* Apaga os displays durante: apagado = período - acendido
*****/

```

```

void EscintilaReloxoDisplays(int8 minutos, int8 segundos, int16 periodo,
int8 ciclo)
{
    int16 acendido = 0;
    int16 apagado = 0;
    acendido = periodo * (ciclo/100);
    apagado = periodo - acendido;
    AmosaReloxoEnDisplays(minutos, segundos);
    delay_ms(acendido);
    ApagaDisplays();
    delay_ms(apagado);
}

/*****
* FUNCIÓN: int8 ModoProgramacion(int8 *minutos_ini, int8 *segundos_ini)
* Permite programar os valores iniciais do segundos e minutos cos botóns:
* - Entra a programar segundos: premendo BOTON1 durante T_PREM_B1
* -- Incrementa segundos: premendo BOTON2 durante T_PREM_B2 ms
* -- Sae da programación de segundos: premendo BOTON1 durante T_PREM_B1 ou
* se non preme ningunha das dúas durante 4s.
* - Entra a programar minutos: ao rematar a programación dos segundos.
* -- Incrementa minutos: premendo BOTON2 durante T_PREM_B2 ms
* -- Sae da programación de minutos: premendo BOTON1 durante T_PREM_B1 ou
* se non preme ningunha das dúas durante 4s.
* RETORNO:
* - return(): false se non se programaron os valores iniciais, tre noutro caso.
* - Nos argumentos da función retórnanse os valores dos minutos e segundos
* iniciais. Se non se programaron, retorna o valor 0. Noutro caso, retorna
* o valor entre 0 e 59 programado en cada caso.
*****/

int8 ModoProgramacion(int8 *minutos_ini, int8 *segundos_ini)
{
    int8 segundos = 0;
    int8 minutos = 0;
    int8 flag_programado = false;
    int8 espera;

    // Detecta a entrada en modo programación
    if (Pulsada(BOTON1, T_PREM_B1) == true)
    {

```

```

flag_programado = true;
// Modo: programación dos segundos.
// Amosa no reloxo -- 00 e esperamos T_PREM_B1 (soltamos BOTON1)
AmosaReloxoEnDisplays(GUION, 0);
delay_ms(T_PREM_B1);
// Se prememos BOTON2, incrementamos segundos
// Se non o prememos durante espera*T_PREM_B2 (40*100ms=4s)pasamos a
// programar minutos
// Se prememos BOTON1, pasamos a programar minutos
espera = 40;
do
{
    if (Pulsada(BOTON2, T_PREM_B2) == true)
    {
        AmosaReloxoEnDisplays(GUION, ++segundos);
        espera = 20;
        if (segundos > 59)
            segundos = 0;
    }
    else
        delay_ms(100);
} while (espera-- > 0 || Pulsada(BOTON1, T_PREM_B1) == false);

// Modo: programación dos minutos.
// Amosa no reloxo 00 -- e esperamos T_PREM_B1 ms (soltamos BOTON1)
AmosaReloxoEnDisplays(0, GUION);
delay_ms(T_PREM_B1);
// Se prememos BOTON2, incrementamos minutos
// Se non o prememos durante espera*T_PREM_B2, saímos de programar o reloxo
// Se prememos BOTON1, saímos de programar o reloxo
espera = 40;
do
{
    if (Pulsada(BOTON2, T_PREM_B2) == true)
    {
        AmosaReloxoEnDisplays(++minutos, GUION);
        espera = 20;
        if (minutos > 59)

```

```

    minutos = 0;
}
else
    delay_ms(100);
} while (espera-- > 0 || Pulsada(BOTON1, T_PREM_B1) == false);
}

// Retornamos os minutos e segundos nos argumentos da función
*segundos_ini = segundos;
*minutos_ini = minutos;
return (flag_programado); }

/*****
* FUNCIÓN: void ReloxoProgramado(int8 minutos_ini, int8 segundos_ini)
* Bucle infinito mostraremos nos 4 displays un reloxo de formato ss mm,
* tomando o valor inicial de minutos_ini y segundos_ini e reiniciándose
* ciclicamente desde 59 59 ata 00 00 (reinicio cada hora).
* NOTA: Para facilitar as probas do programa, podemos cambiar o define
* VELOCIDADE de 1000 (segundos) a 100 (10 veces máis rápido)
* NOTA: A función non é un contador preciso xa que non ten en conta o tempo
* de execucións das instrucións (4xfosc)
*****/

void ReloxoProgramado(int8 minutos, int8 segundos)
{ // Bucle infinito do reloxo
    while (true) {
        // Bucle dos minutos, reiniciado ao pasar de 59
        while(minutos < 60) {
            // Bucle dos segundos, reiniciado ao pasar de 59
            while (segundos < 60) {
                AmosaReloxoEnDisplays(minutos, segundos);
                // Esperamos os milisegundos indicados no define VELOCIDADE
                // 1000 é o normal para contar segundos
                // é dez veces máis rápido para comprobar o programa.
                delay_ms(VELOCIDADE);
                segundos++; // Incrementamos o contador de segundos
            }
            // Reinicio do contador de segundos cada 60 segundos (de 00 a 59 en ss)
            segundos = 0;
            minutos++; // Incrementamos o contador de minutos
        }
    }
}

```

```

// Reinicio do contador de minutos cada 60 minutos (de 00 a 59 en mm)
minutos = 0;
}}

/*****
* FUNCIÓN: void AmosaReloxoEnDisplays(int8 minutos, int8 segundos)
* Recibe minutos (ente 00 e 59) y segundos (entre 00 e 59) para amosalos nos
* 4 displays da placa, no formato: mm ss.
* - Se recibe GUION nos minutos ou nos segundos, amosa no seu lugar "--".
* - Garda nun vector cada díxito decimal, minutos e segundos.
* - Escribe nos displays os díxitos usando os rexistros de desprazamento.
* Cada díxito decimal é codificado ao valor requirido para amosalo no display
* A función write_expanded_outputs(*int eo) está defina no include do
* driver 74595.c.
* - Nos displays amosa do 00 00 ao 59 59.
*****/

void AmosaReloxoEnDisplays(int8 minutos, int8 segundos)
{
    int8 amosa_dixitos[4];
    // Calcula cada díxito decimal e obtén o seu valor codificado vector dixitos[]
    // Copia o resultado no vector amosa_dixitos[] que pasará á función do
    // driver 74595.c para mostralos nos displays.
    if (minutos == GUION)
        amosa_dixitos[0] = amosa_dixitos[1] = 0x40; // Aмоса "--"
    else
    {
        amosa_dixitos[0] = dixitos[minutos / 10 % 10];
        amosa_dixitos[1] = dixitos[minutos % 10];
    }
    if (segundos == GUION)
        amosa_dixitos[2] = amosa_dixitos[3] = 0x40; // Aмоса "--"
    else
    {
        amosa_dixitos[2] = dixitos[segundos / 10 % 10];
        amosa_dixitos[3] = dixitos[segundos % 10];
    }
    write_expanded_outputs(amosa_dixitos);
}

/*****

```

* FUNCIÓN: ApagaDisplays()

* Apaga os displays activando o terceiro estado na saída dos rexistros 74595

* (alta impedancia) poñendo nivel HIGH nas entradas *OE dos rexistros 74595.

*****/

```
void ApagaDisplays()
```

```
{
```

```
    output_high(EXP_OUT_ENABLE);
```

```
}
```

*****/

* FUNCIÓN: int8 Pulsada(int8 entrada, int8 tempo)

* Determina si a entrada permanece premida (valor high nos botóns)

* o tempo indicado (tempo dado para evitar os rebotes ou incrementar o control)

* RETORNO:

* - return(): true ou false.

*****/

```
int8 Pulsada(int8 entrada, int8 tempo)
```

```
{
```

```
    if (input(entrada) == ENTRADA_H)
```

```
    {
```

```
        delay_ms(tempo); // Repite a medida para evitar rebotes: tempo (miliseg)
```

```
        if (input(entrada) == ENTRADA_H)
```

```
            return(true); }
```

```
    return(false); }
```

2.2.6 LETURA DE TEMPERATURA BAIXO I²C EN LCD E DISPLAY

```
/******  
* FICHEIRO: TemperaturaI2C.c  
* CONTIDO: Programa sobre o PIC16F877A que mostra o uso do bus I2C para ler  
* a temperatura medida polo CI DS1631 (Termómetro) e presentala de dous formas:  
* - Displays 7-segmentos  
* - LCD 2x16 caracteres  
* FUNCIONES:  
* void main(): .  
*****/  
  
/****** CONFIGURACIÓN ESPECÍFICA DA PLACA OU DO EXERCICIO:  
* Frecuencia de oscilación do PIC: clock=10000000  
* Pins e número dos rexistros de desprazamento (CI 74595): EXP_OUT_DO,  
* EXP_OUT_CLOCK, EXP_OUT_ENABLE, NUMBER_OF_74595  
* Pins do bus I2C: scl=PIN_C3, sda=PIN_C4  
* Dirección I2C do termómetro (CI DS1631): DIR_I2C_TERM_W, DIR_I2C_TERM_R  
* Pins do LCD: LCD_DB4, LCD_DB5, LCD_DB6, LCD_DB7, LCD_RS, LCD_RW e  
LCD_E  
*****/  
  
// Contorno básico do hardware  
#include <16f877A.h> // Usamos PIC16F877A  
#device ICD=TRUE // Usamos depurador ICD  
#fuses HS,NOLVP,NOWDT,PUT // Non aceptamos programación nivel baixo  
#use delay (clock=10000000) // Frecuencia de oscilación do PIC: 10Mhz  
  
/******  
* Fundamento: Amosa díxitos nos 4 displays mediante o driver: ../Drivers/74595.c  
*****/  
  
// Defines e include do driver da configuración de rexistros 74595  
#define EXP_OUT_DO PIN_D1 // SER (Dato de entrada serie)  
#define EXP_OUT_CLOCK PIN_D0 // SCLK (Reloxo, flanco ascendente. Entra dato)  
#define EXP_OUT_ENABLE PIN_D2 // RCLK (Reloxo, flanco ascendente. Dato ao  
LATCH)  
  
// e *OE (LOW: Habilita a saída do dato paralelo)  
#define NUMBER_OF_74595 4 // Temos 4 displays e, polo tanto, 4 rexistros  
#include <74595.c> // Include do ../Driver/74595.c  
  
// Codificación dos díxitos decimais do '0' ao '9' cara a activar os displays
```

```
const int8 dixitos[10] = {0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};
```

```
/******
```

* Fundamento: Xestión de escritura no LCD con 4 pins de datos. Engadimos no

* directorio de drivers: ../Driver/LCD_AC162D.c

* Definimos os pins do noso montaxe.

* No utilizamos a lectura do LCD.

```
*****/
```

```
#define LCD_DB4 PIN_C5
```

```
#define LCD_DB5 PIN_D3
```

```
#define LCD_DB6 PIN_D5
```

```
#define LCD_DB7 PIN_D7
```

```
#define LCD_RS PIN_E1
```

```
#define LCD_RW PIN_E2
```

```
#define LCD_E PIN_A2
```

```
#include <LCD_AC162D.c>
```

```
/******
```

* Fundamento: Configuramos o PIC como Master, indicamos os pins de CLOCK e DATA

* do bus e deixamos por defecto a velocidade baixa do bus: 100Khz

* O termómetro (CI DS1631) traballará como Slave, coa dirección dada polo

* CI (4 primeiros bits fixos: 1.0.0.1) e a implementación na placa (A2.A1.A0

* están a GND, configurando os seguintes 3 bits. O último bit da dirección

* indica a dirección da transferencia do dato. Así:

* A dirección I2C do termómetro é un byte de 8 bits: 1.0.0.1.A2.A1.A0.RW;

* Os 4 primeiros bits A2,A1,A0 están a GND

* (transferencia de datos do Master ao Slave)W=0, W=10010000=0x90

* (transferencia de datos do Slave ao Master)R=1. R=10010001=0x91

* O rexistro donde o termómetro garda a temperatura medida ten dous bytes pero

* 12bits (os 4 menos significativos son 0). O bit máis significativo indica o

* signo; noso caso traballaremos só con temperaturas positivas (bit 0) e

* valores enteiros codificados en binario natural no byte máis significativo.

* No caso dunha temperatura negativa, amosaremos un 0.

```
*****/
```

```
// Configuración do bus I2C
```

```
#use i2c(master, scl=PIN_C3, sda=PIN_C4)
```

```
// Direccións do termómetro para transferencias de datos
```

```
#define DIR_I2C_TERM_W 0x90
```

```
#define DIR_I2C_TERM_R 0x91
```

```
// Órdenes usadas cara ao control do Termómetro (CI DS1631, Commands)
```

```

#define ORD_INI_CONV_TEMP 0x51 // Orden de iniciar conversións de temperaturas
#define ORD_R_TEMP 0xAA // Orden de lectura dos dous bytes da temperatura
#define ORD_W_REG 0xAC // Orden de escritura do rexistro (1byte)

/*****

* FUNCIÓN: void AmosaEnDisplays(int16 numero)
* Recibe un número entre 0 e 9999 para amosalo nos 4 displays da placa.
* - Garda nun vector cada díxito decimal.
* - Escribe nos displays os díxitos usando os rexistros de desprazamento.
* Cada díxito decimal é codificado ao valor requirido para amosalo no display
* A función write_expanded_outputs(*int eo) está defida no include do
* driver 74595.c.
* - Nos displays amosa do 0000 ao 9999
*****/

void AmosaEnDisplays(int16 numero)
{
    int8 amosa_dixitos[4];
    // Calcula cada díxito decimal e obtén o seu valor codificado vector dixitos[]
    // Copia o resultado no vector amosa_dixitos[] que pasará á función do
    // driver 74595.c para mostralos nos displays.
    amosa_dixitos[0] = dixitos[numero / 1000 % 10];
    amosa_dixitos[1] = dixitos[numero / 100 % 10];
    amosa_dixitos[2] = dixitos[numero / 10 % 10];
    amosa_dixitos[3] = dixitos[numero % 10];
    write_expanded_outputs(amosa_dixitos);
}

/*****

* FUNCIÓN: void main()
*****/

void main()
{
    int temp_high, temp_low;
    char buffer[10];
    int8 caracter;
    lcd_init();
    i2c_start();
    i2c_write(DIR_I2C_TERM_W); // Selección do termómetro pola súa dirección (W)

```

```

i2c_write(ORD_W_REG); // Orden de escrita no rexistro de configuración
i2c_write(0x0C); //Rexistro = 00001100 = Resolución 12 bits (750ms)
i2c_start();
i2c_write(DIR_I2C_TERM_W); // Selección do termómetro pola súa dirección (W)
i2c_write(ORD_INI_CONV_TEMP); //Orden de iniciar a conversión de temperaturas
i2c_stop();
    delay_ms(750); //Tempo antes da primeira lectura dunha temperatura válida
    // A función de escritura de carácter no LCD permite transmitir unha cadea
    // de caracteres sempre que esta definida no tempo de compilación.
    lcd_putc("Temperatura: ");
    while(true)
    {
        i2c_start();
        i2c_write(DIR_I2C_TERM_W); // Selección do termómetro pola súa dirección (W)
        i2c_write(ORD_R_TEMP); // Orden de lectura da temperatura
        i2c_start();
        i2c_write(DIR_I2C_TERM_R); // Selección do termómetro pola súa dirección (R)
        temp_high = i2c_read(1); // Le o MSB do dato da temperatura
        temp_low = i2c_read(0); // Le o LSB do dato da temperatura
        i2c_stop();
        // No caso de recibir temperatura negativa, amosamos un 0
        if (temp_high > 127)
            temp_high = 0;
        AmosaEnDisplays(temp_high);
        // Para amosar un conxunto de caracteres no LCD temos que facelo carácter
        // a carácter. O facemos mediante buffer[].
        sprintf(buffer, "%d", temp_high);

        lcd_gotoxy(14,1); // Posición 14, despois da cadea "Temperatura: "
        character = buffer[0];
        lcd_putc(character);
        character = buffer[1];
        lcd_putc(character);
        lcd_putc(0xDF); // Simbolo de grados
        delay_ms(2000); // Espera 2s antes de iniciar unha nova lectura. Basta 750.
    }
}

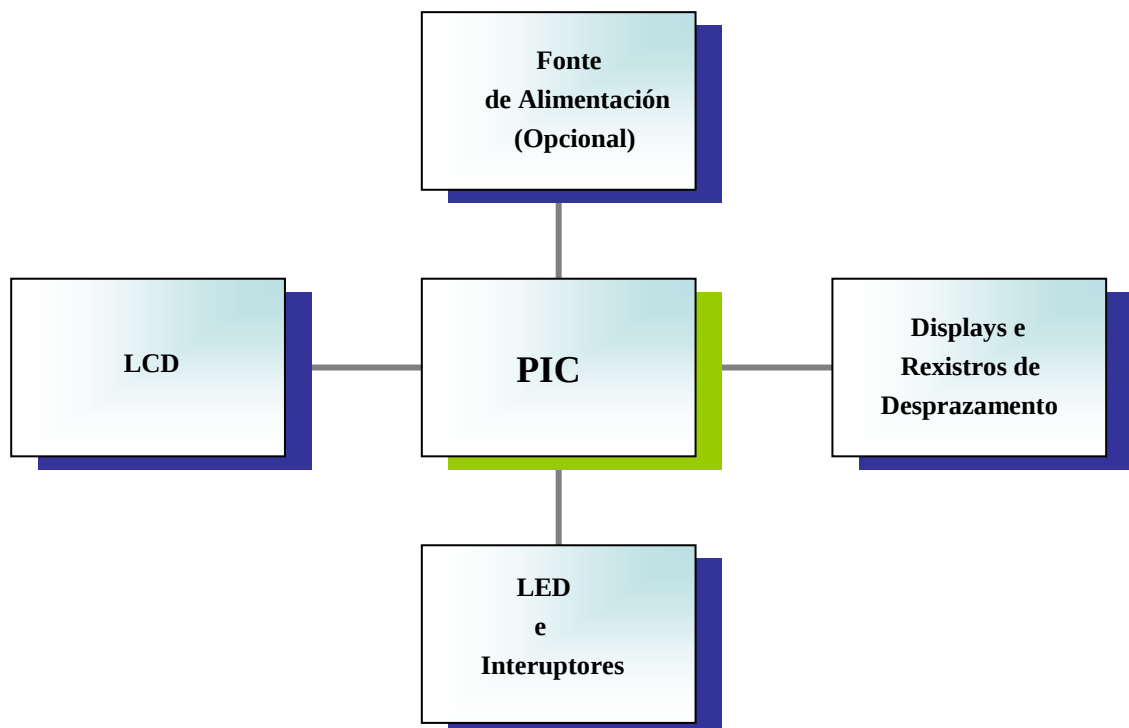
```

2.3 Construción dun Programador de PIC's co OrCad

O deseño da placa para realizar un adestrador de PIC's esixe por parte do alumnado un amplo estudo das características técnicas dos diferentes compoñentes que forman parte dela. O funcionamento básico destes compoñentes foi estudado noutros módulos como son Electrónica Dixital e Microprogramable, para os compoñentes dixitais, e Electrónica Analóxica, para os compoñentes analóxicos. Así mesmo entendemos que o alumnado posúe a destreza necesaria no manexo do programa OrCad para desenvolver as distintas fases do proxecto.

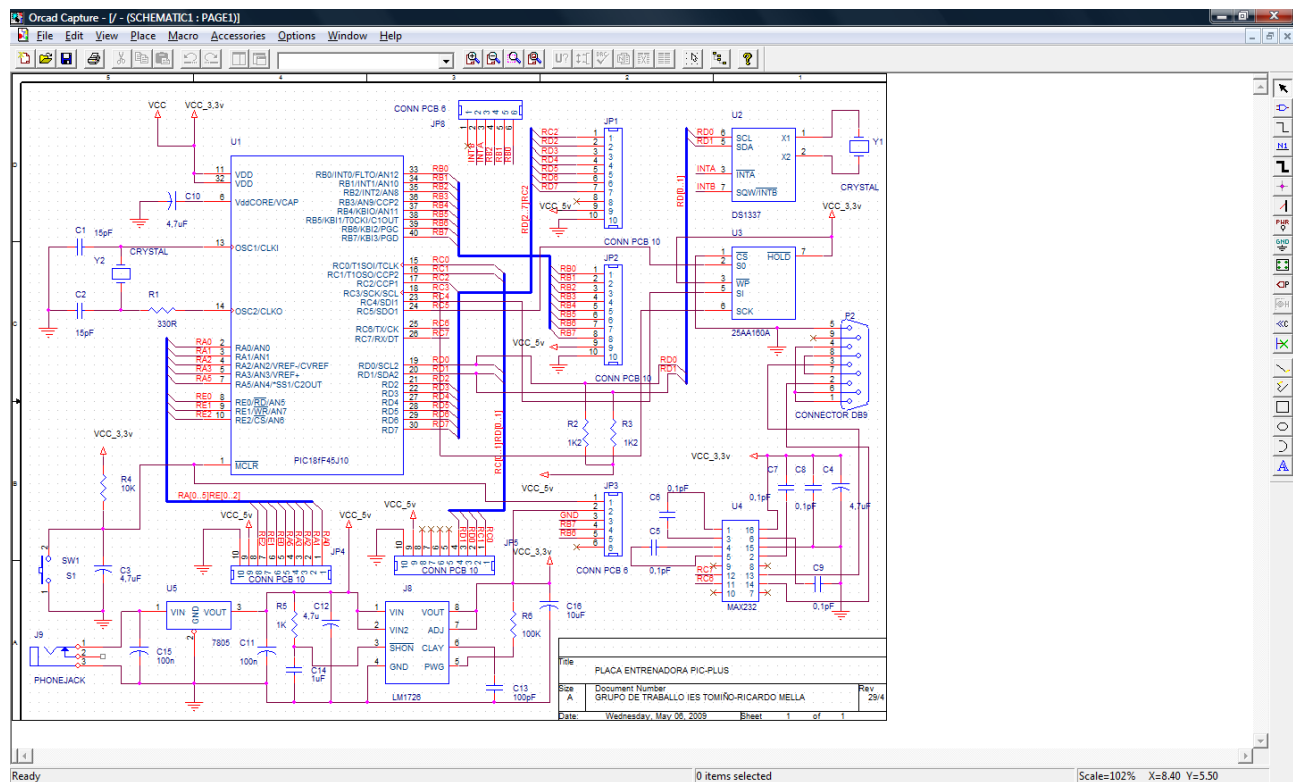
2.3.1 Previo o deseño co programa OrCad

O primeiro paso antes de abordar o deseño debe ser o estudo das follas características dos distintos compoñentes que van formar parte do adestrador para comprender o seu funcionamento, descrições das patillas dos integrados, diferentes tipos de encapsulados, dimensións dos compoñentes, tipos de conectores a usar, etc., así como da distribución dos compoñentes nas diferentes placas de CI. Engadimos unha carpeta cos datasheets necesarios para esta parte inicial. Un esquema de bloques de cada unha das placas axuda para iniciar o deseño de cada unha delas co programa OrCad segundo podemos ver na seguinte figura. Segundo este deseño é posible ampliar o adestrador con distintas placas como poden ser: Control de potencia con optotriacs, relés, sensores de presión, etc.



2.3.2 Deseño co programa OrCad Capture. Placa do PIC

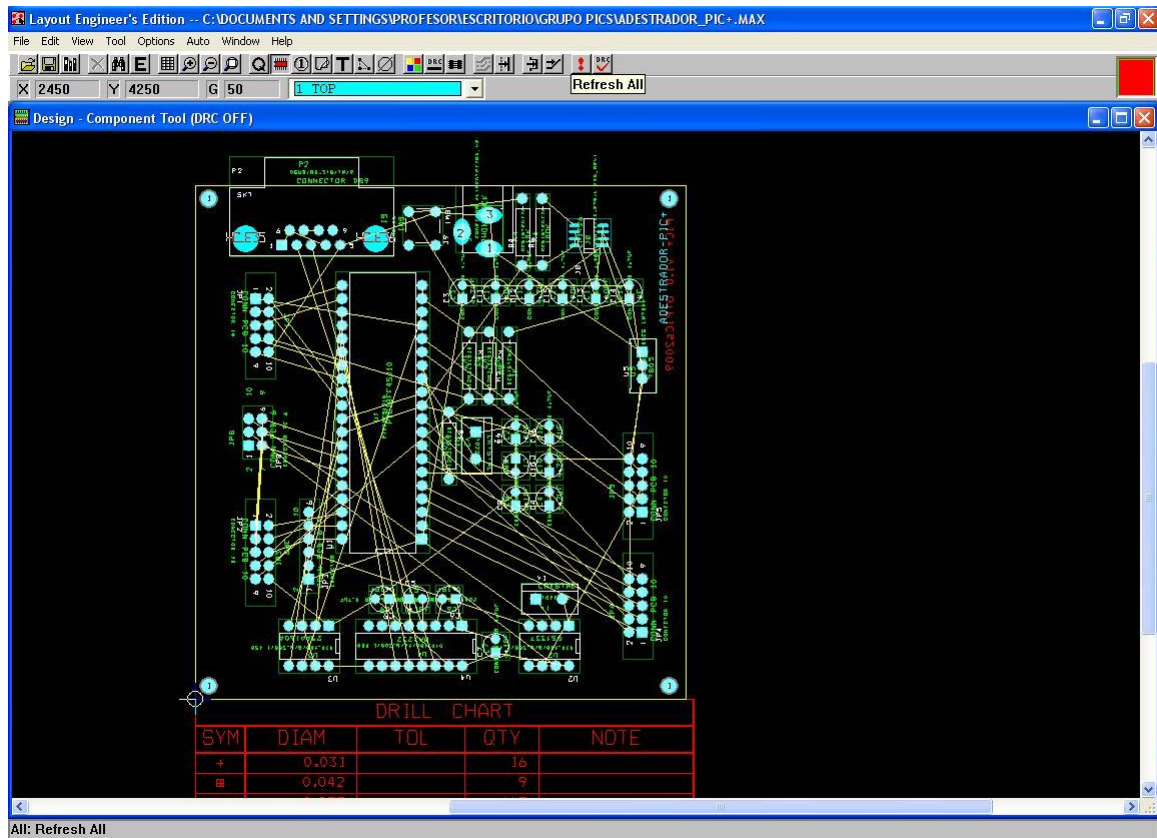
Abrimos o programa Capture e iniciamos un novo proxecto (seleccionamos o nome do proxecto e a carpeta onde o imos gardar) colocando todos e cada un dos distintos compoñentes que forman esta placa. No caso de que non atopemos algún deles nas librerías debemos facelo nos, como é o caso dos displays 7 segmentos, ou adaptar un da librería. Realizamos as conexións entre tódolos compoñentes e obtemos un esquema final da placa tal e como aparece na seguinte figura:



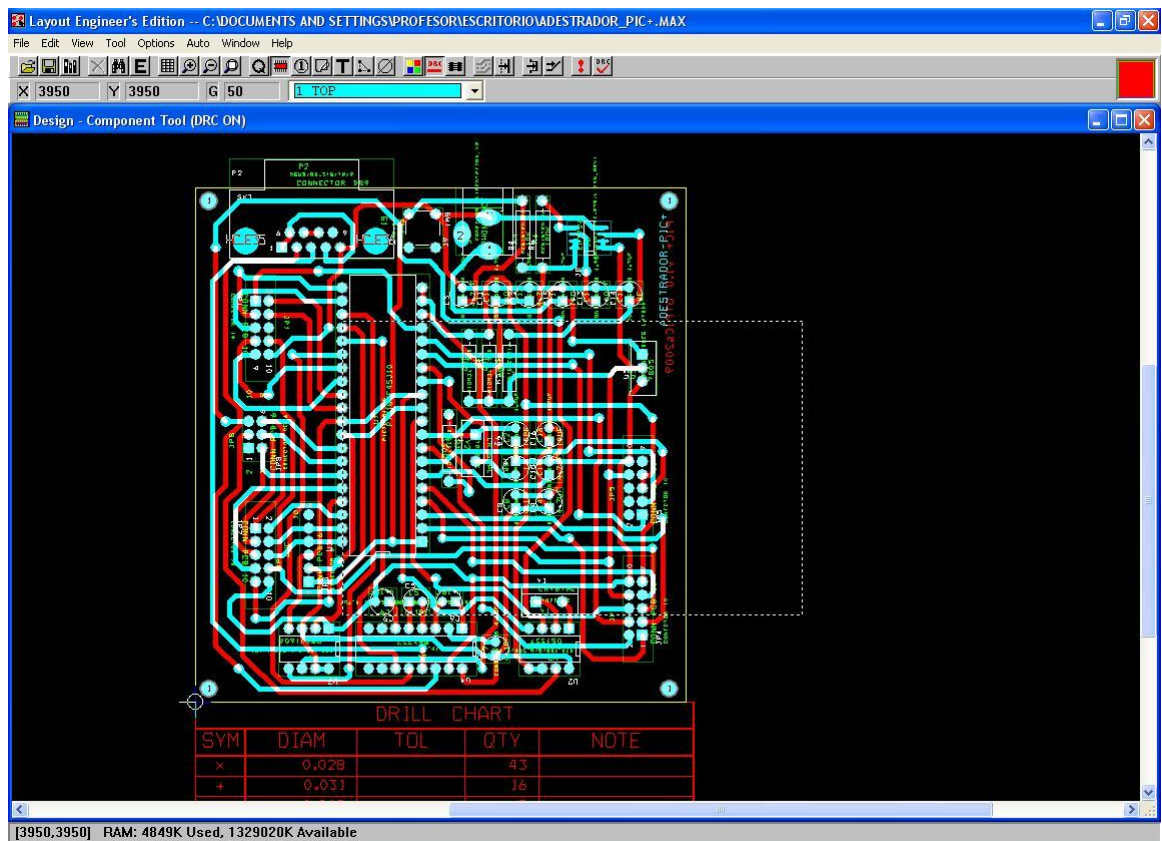
Comprobamos que tódolos compoñentes estean conxicionados adecuadamente coa ferramenta “Design Rules Check”, creamos a lista de materiais necesarios coa ferramenta “Bill of Materials” e finalmente creamos o ficheiro necesario para o programa OrCad Layout coa ferramenta “Create Netlist”.

2.3.3 Deseño co programa OrCad Layout. Placa do PIC

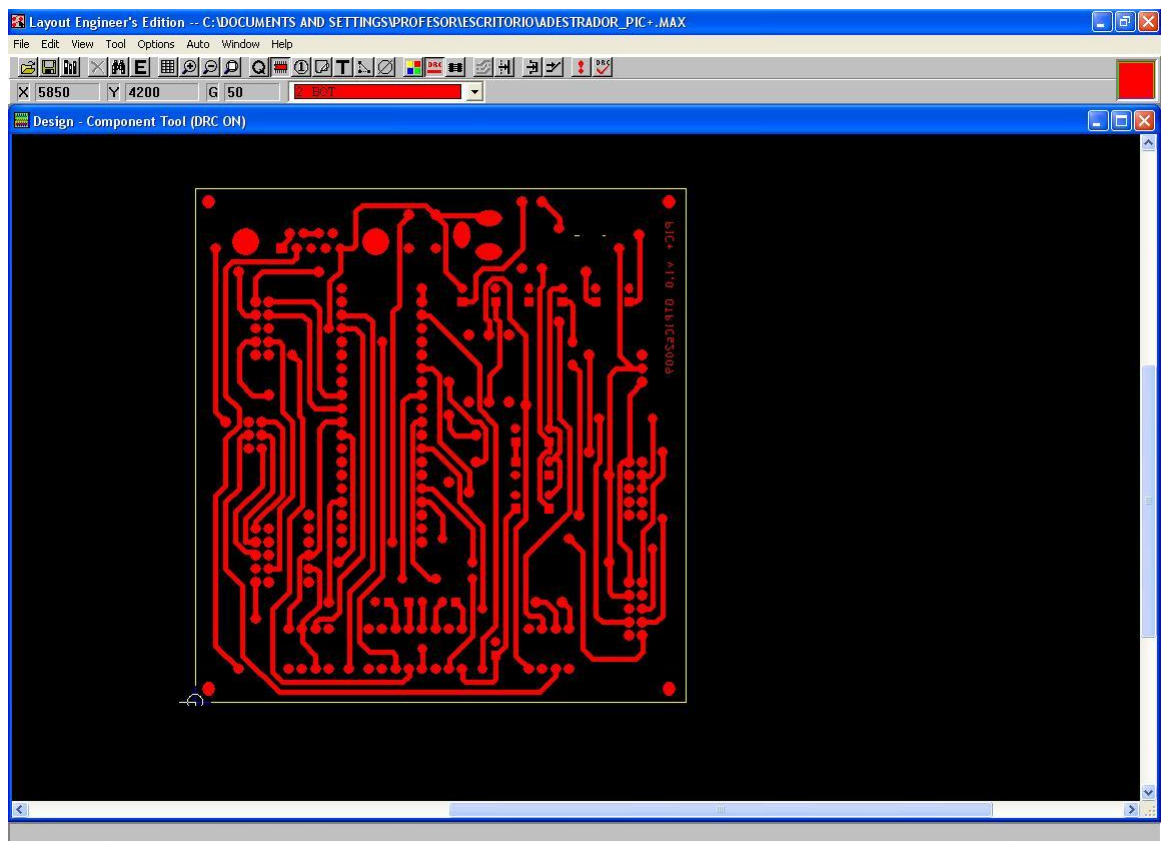
Abrimos o programa Orcad Layout e coa opción File → New seleccionamos o ficheiro de Tecnoloxía metric.tch. Engadimos o ficheiro de netlist creado co Orcad Capture anteriormente e seleccionamos a carpeta onde gardaremos o ficheiro Layout max file. Buscamos a mellor disposición dos compoñentes para o proceso de rotado da placa e obtemos o seguinte resultado:



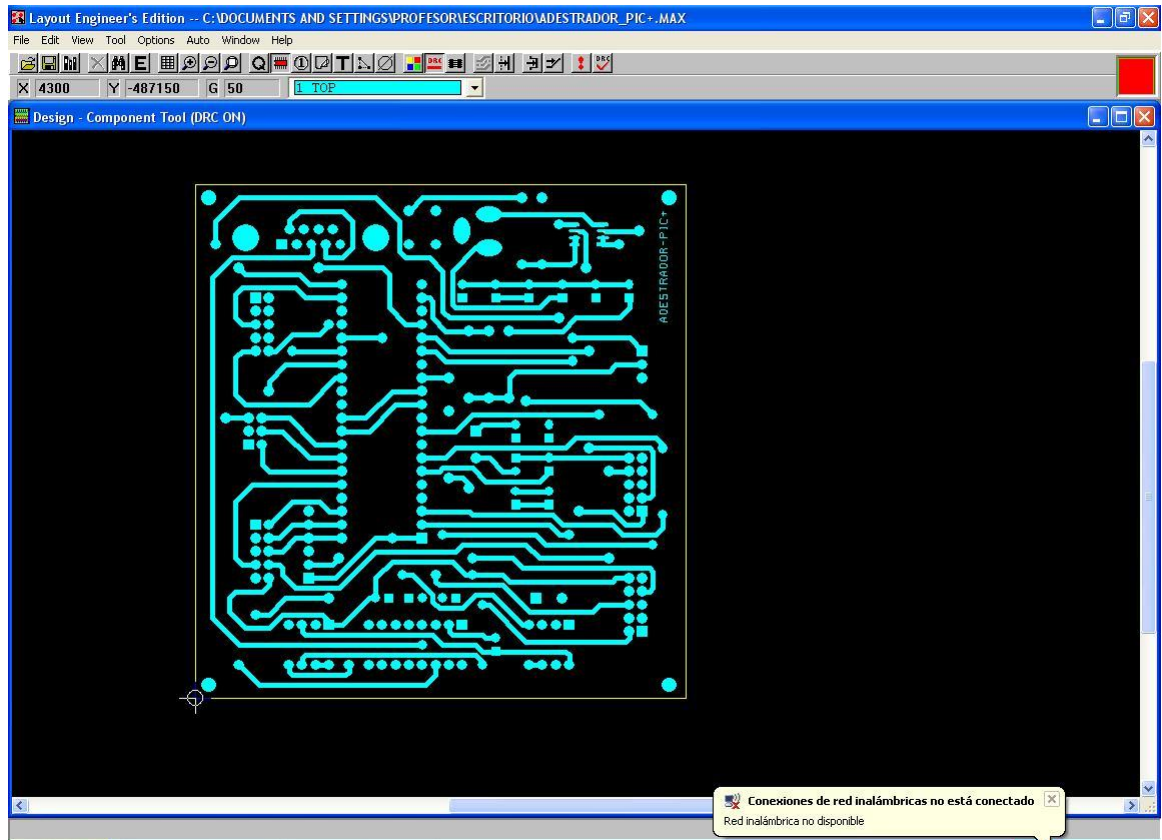
Procedemos ao proceso de rotado automático a dobre cara da placa, obtendo o deseño que aparece na seguinte figura:



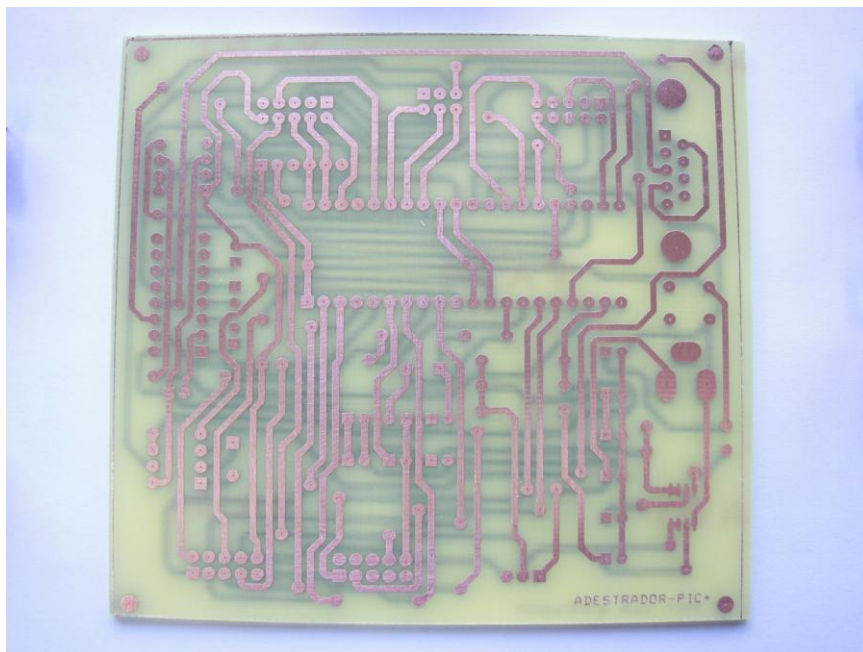
Nas seguintes imaxes podemos ver a cara de pistas:

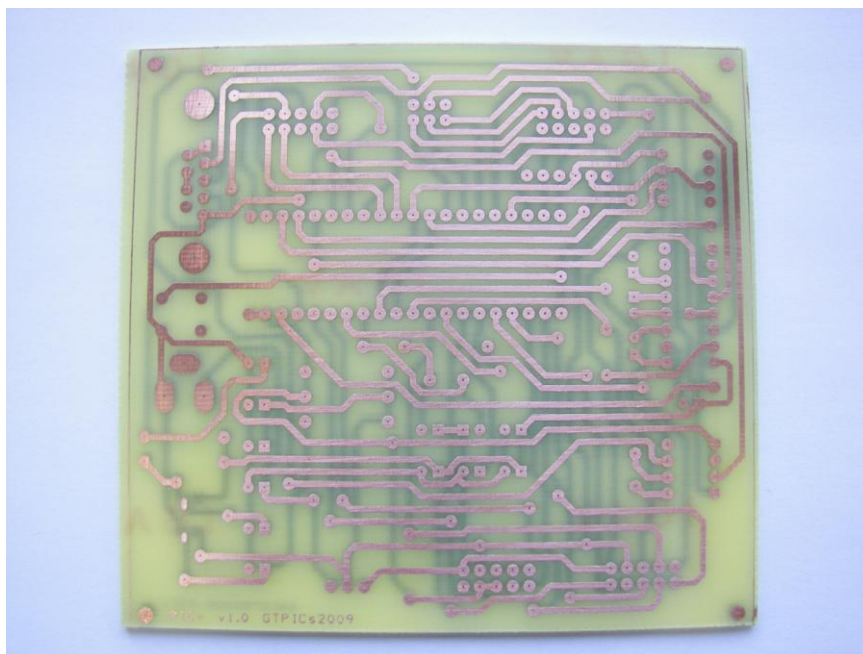


e de compoñentes:



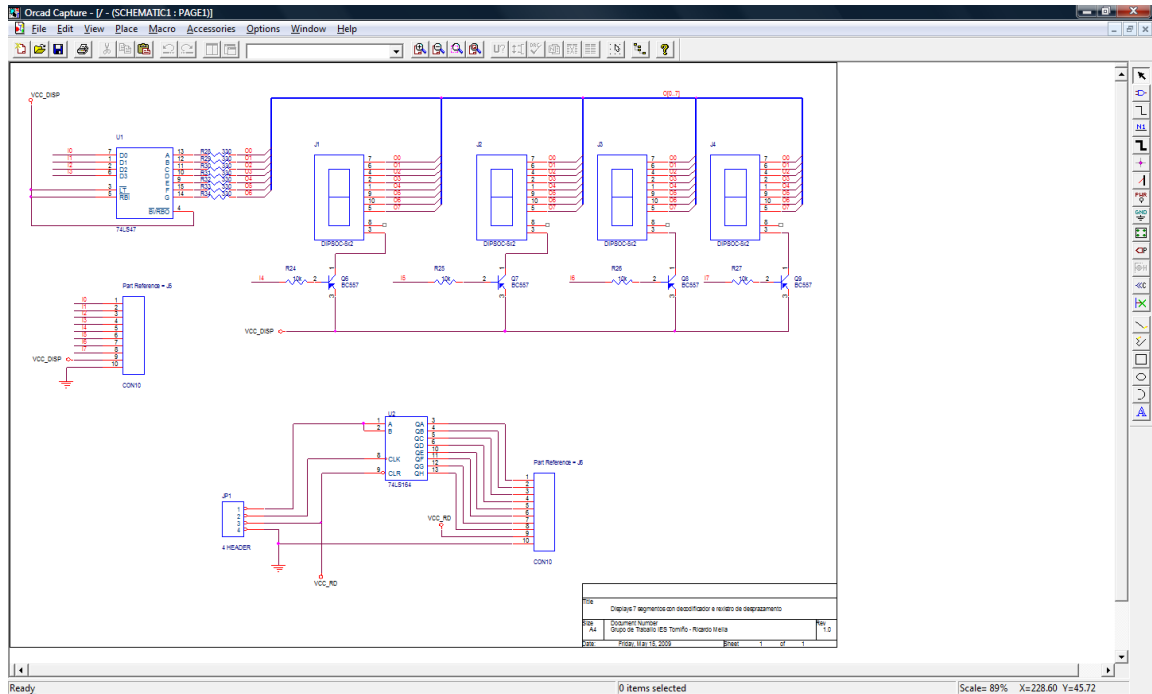
O seguinte proceso é a impresión destes esquemas en papel vexetal. Imos utilizar placa fotosensible de dobre cara. Debemos ter a precaución para o proceso de revelado que os fotolitos coincidan perfectamente e non teñamos erros na placa final. Unha vez dispostos os fotolitos coa placa na insoladora (ten que ser de dobre cara), procedemos ao proceso de insolado axustando o tempo de exposición (podemos facer varias probas ata atopar o tempo correcto). Revelamos a placa e procedemos ao atacado e o resultado que obtemos pódese ver nas dúas seguintes figuras:





2.3.4 Deseño co programa OrCad Capture. Placa dos Displays e Rexistros de Desprazamento

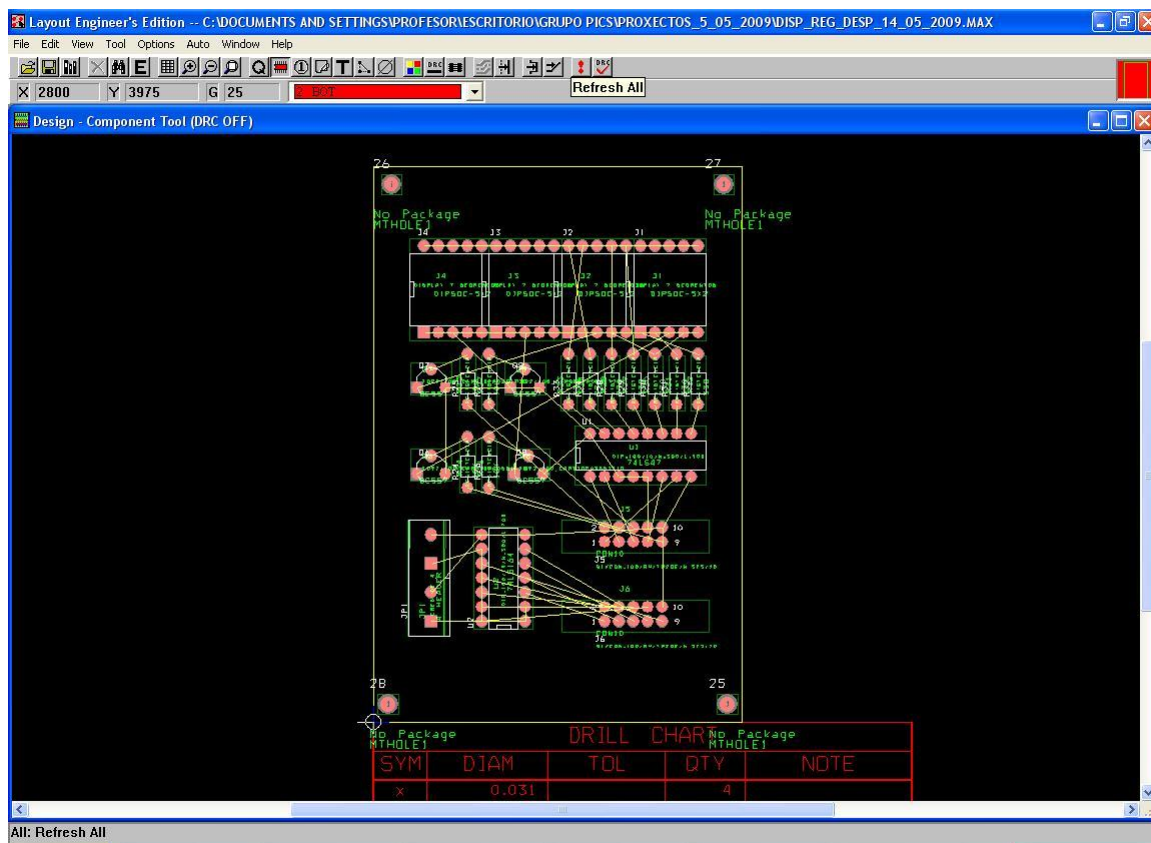
Abrimos o programa Capture e iniciamos un novo proxecto (seleccionamos o nome do proxecto e a carpeta onde o imos gardar) colocando todos e cada un dos distintos compoñentes que forman esta placa. No caso de que non atopemos algún deles nas librerías debemos facelo nos, como é o caso dos displays 7 segmentos, ou adaptar un da librería. Realizamos as conexións entre tódolos compoñentes e obtemos un esquema final da placa tal e como aparece na seguinte figura:



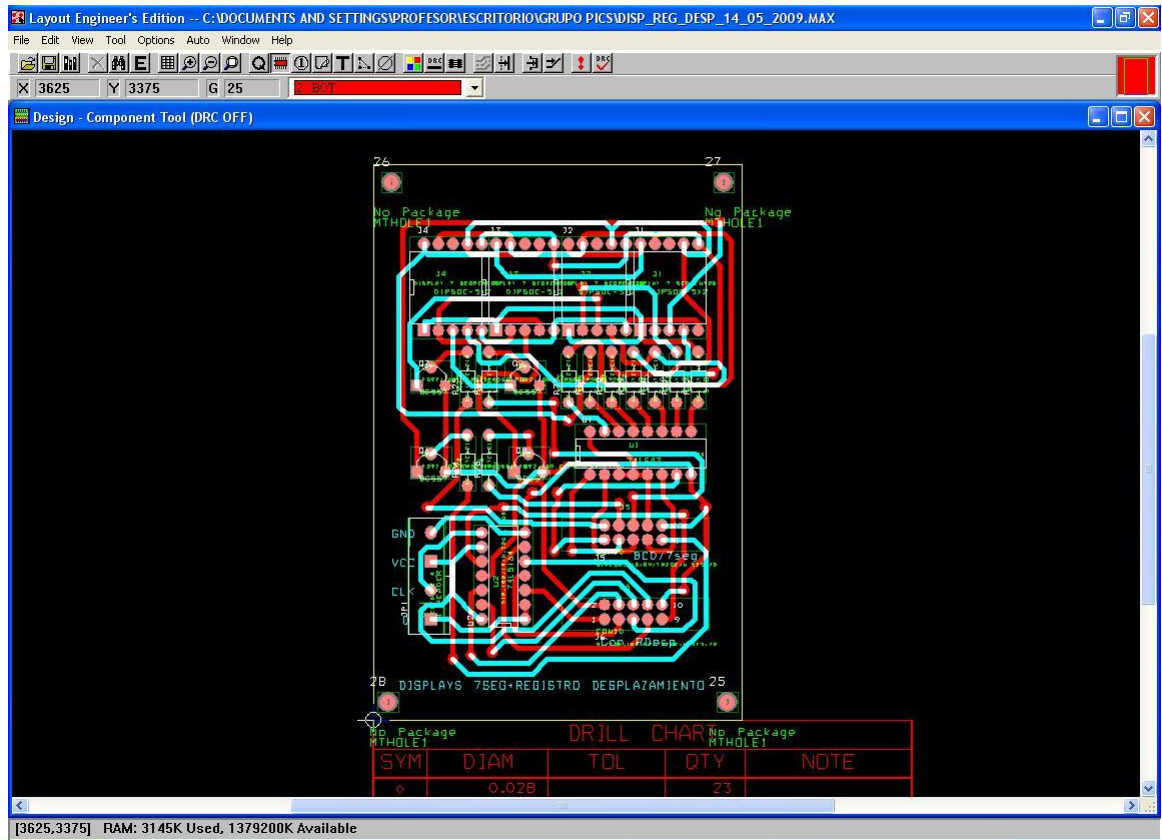
Comprobamos que tódolos compoñentes estean connexionados adecuadamente coa ferramenta “Design Rules Check”, creamos a lista de materiais necesarios coa ferramenta “Bill of Materials” e finalmente creamos o ficheiro necesario para o programa OrCad Layout coa ferramenta “Create Netlist”.

2.3.5 Deseño co programa OrCad Layout. Placa dos Displays e Rexistros de Desprazamento

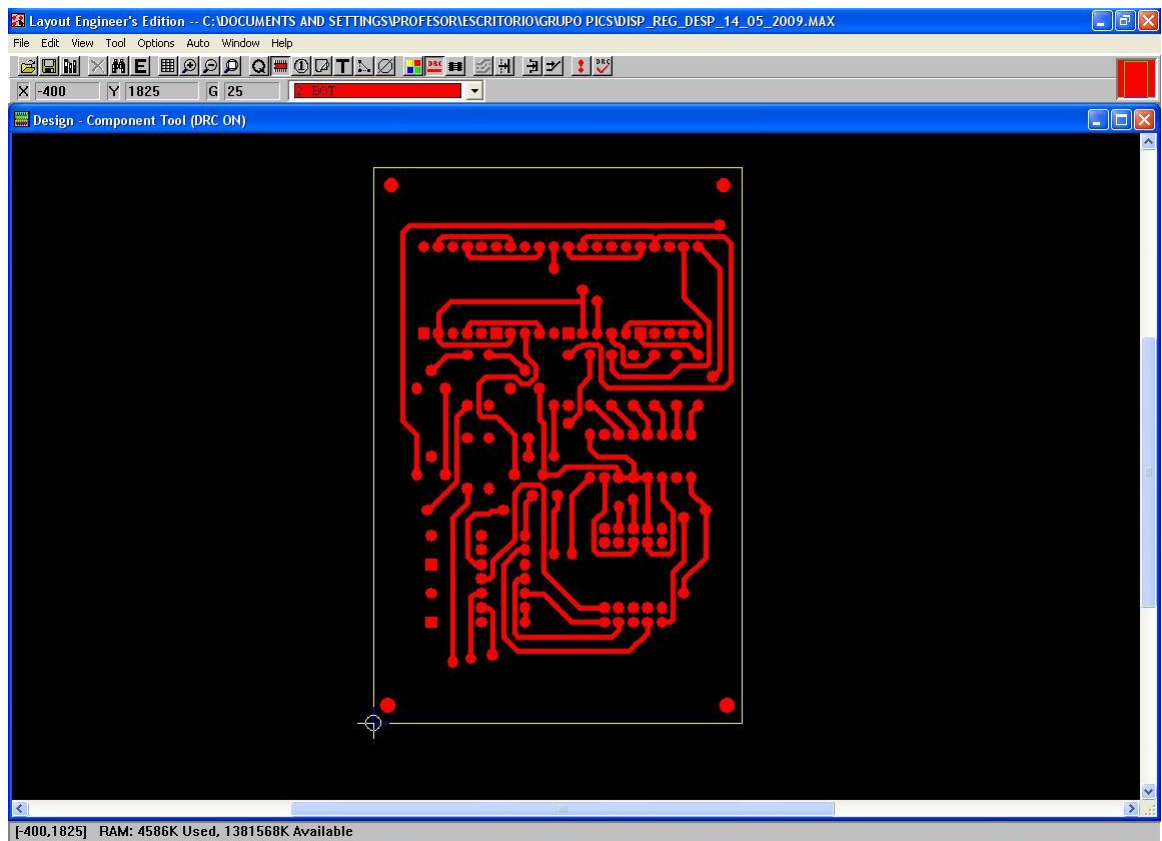
Abrimos o programa Orcad Layout e coa opción File → New seleccionamos o ficheiro de Tecnoloxía metric.tch. Engadimos o ficheiro de netlist creado co Orcad Capture anteriormente e seleccionamos a carpeta onde gardaremos o ficheiro Layout max file. Buscamos a mellor disposición dos compoñentes para o proceso de rotado da placa e obtemos o seguinte resultado:



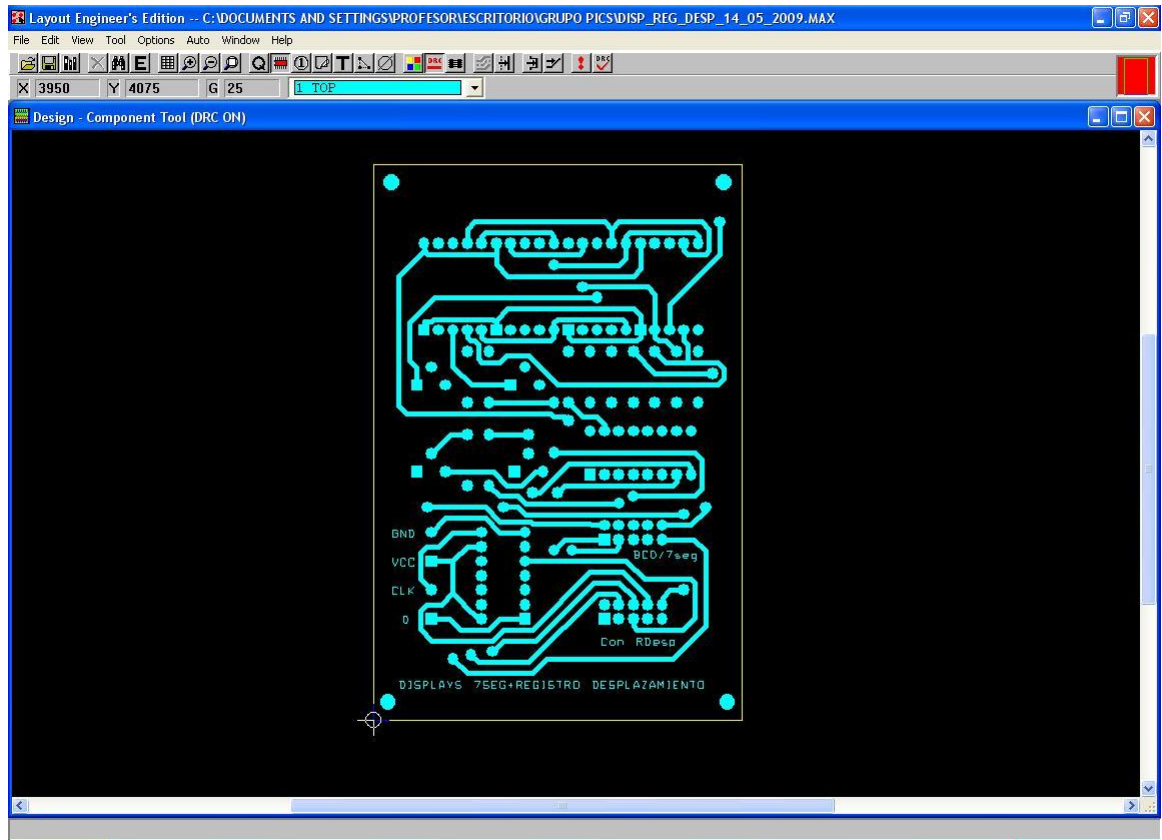
Procedemos ao proceso de rotado automático a dobre cara da placa, e o deseño obtido é:



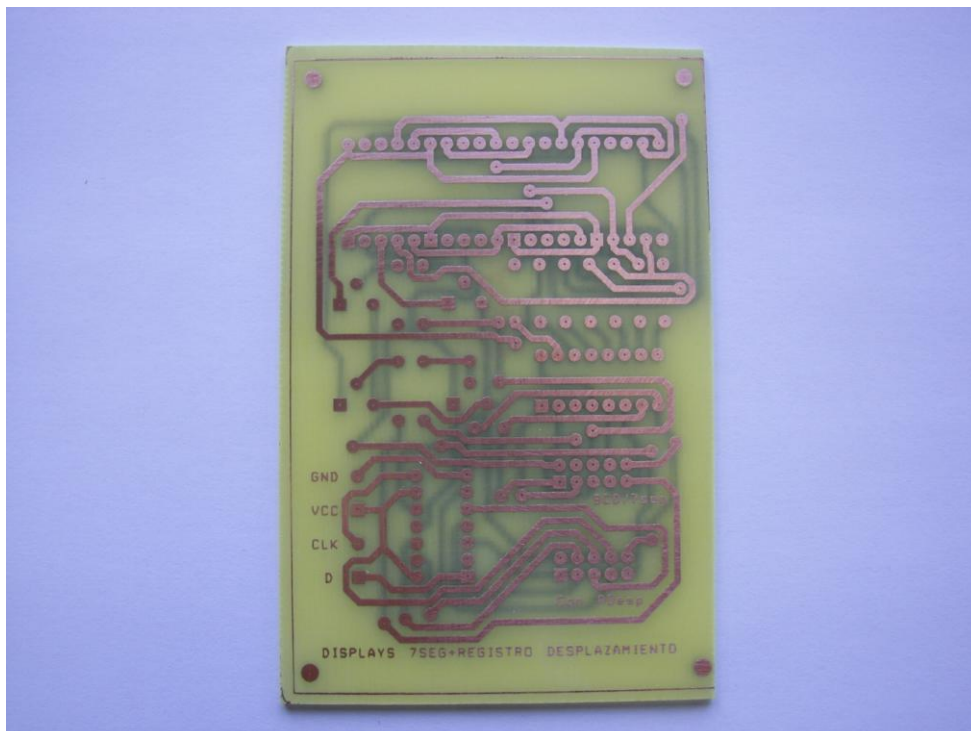
Nas seguintes imaxes podemos ver a cara de pistas:

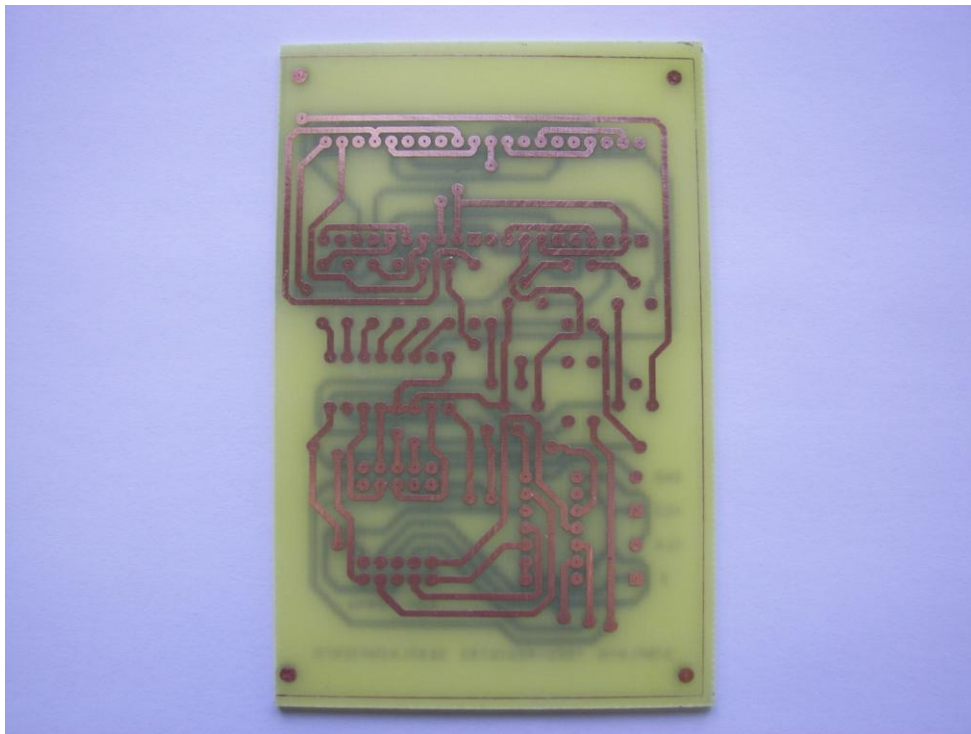


e de compoñentes:



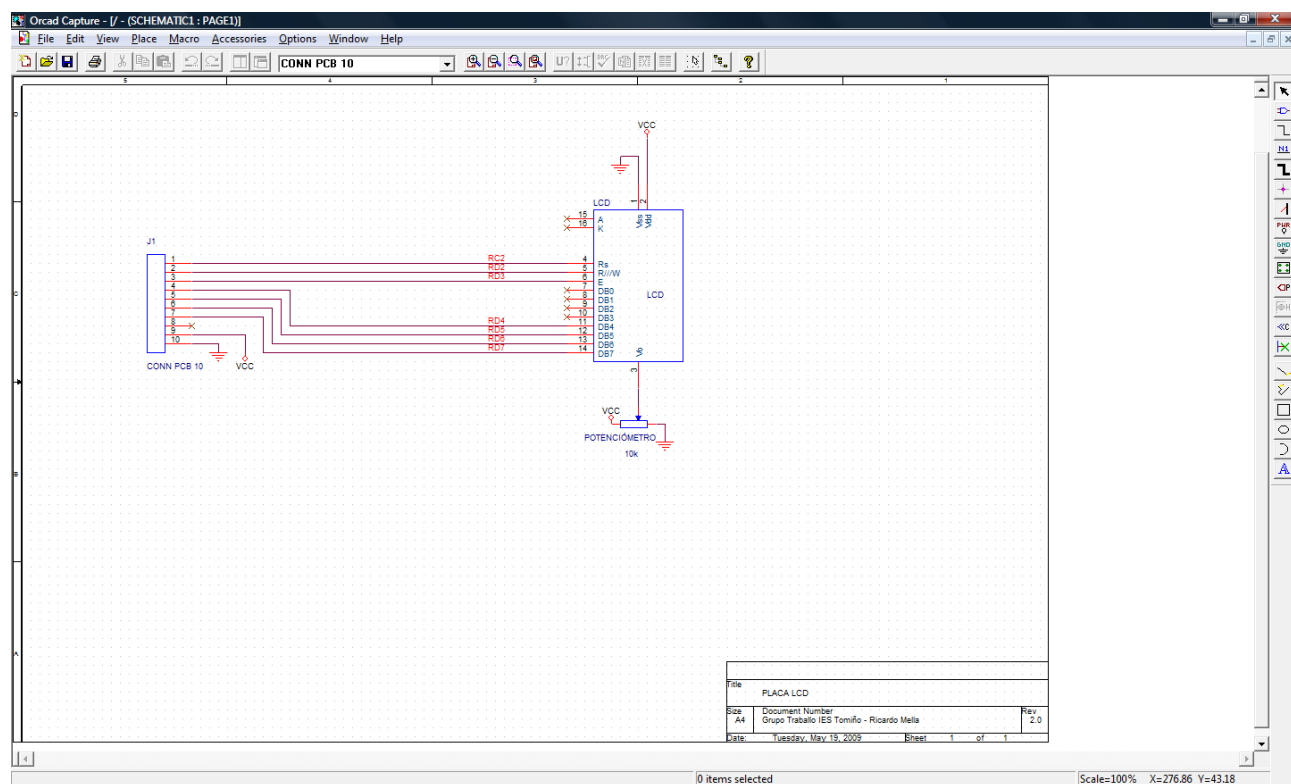
O seguinte proceso é a impresión destes esquemas en papel vexetal. Imos utilizar placa fotosensible de dobre cara. Debemos ter a precaución para o proceso de revelado que os fotolitos coincidan perfectamente e non teñamos erros na placa final. Unha vez dispostos os fotolitos coa placa na insoladora (ten que ser de dobre cara), procedemos ao proceso de insolado axustando o tempo de exposición (podemos facer varias probas ata atopar o tempo correcto). Revelamos a placa e procedemos ao atacado obtendo un resultado como o que vemos nas dúas figuras seguintes:





2.3.6 Deseño co programa OrCad Capture. Placa do LCD

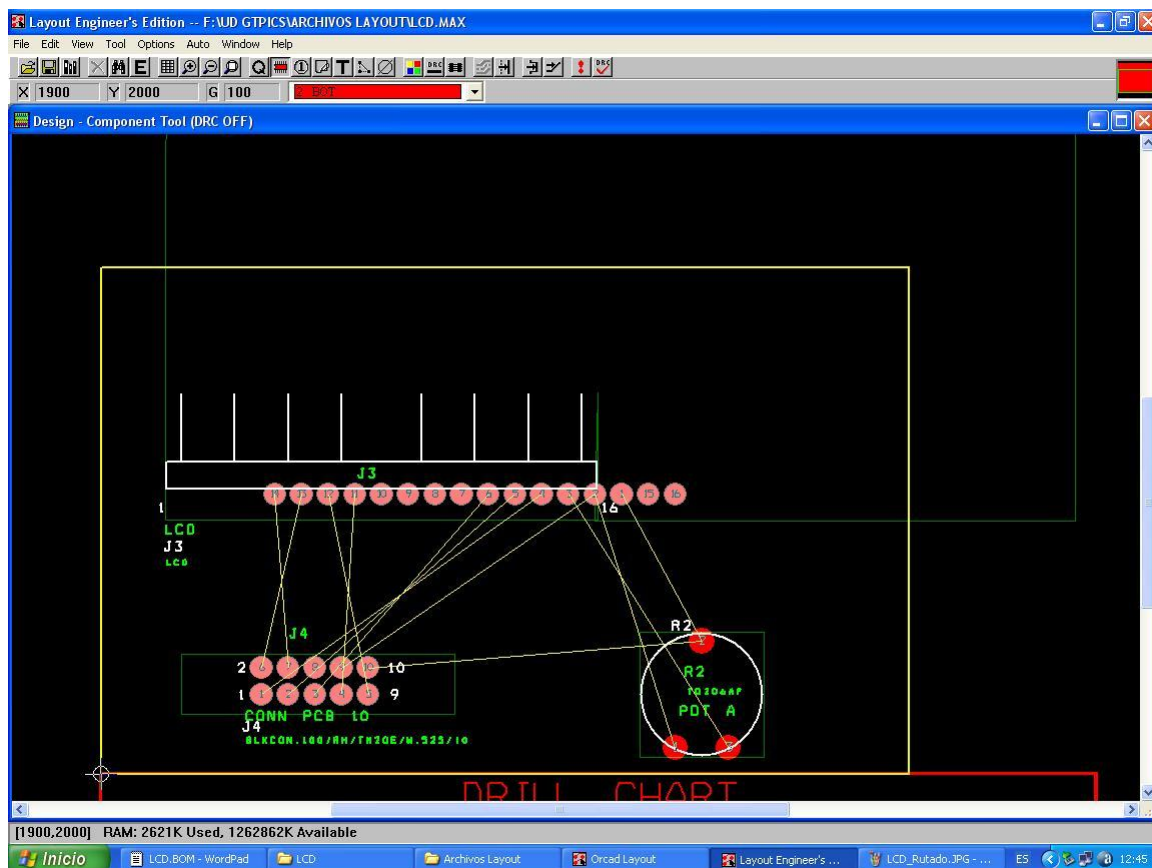
Abrimos o programa Capture e iniciamos un novo proxecto (seleccionamos o nome do proxecto e a carpeta onde o imos gardar) colocando todos e cada un dos distintos compoñentes que forman esta placa. No caso de que non atopemos algún deles nas librerías debemos facelo nos. Realizamos as conexións entre tódolos compoñentes e obtemos un esquema final da placa tal e como aparece na seguinte figura:



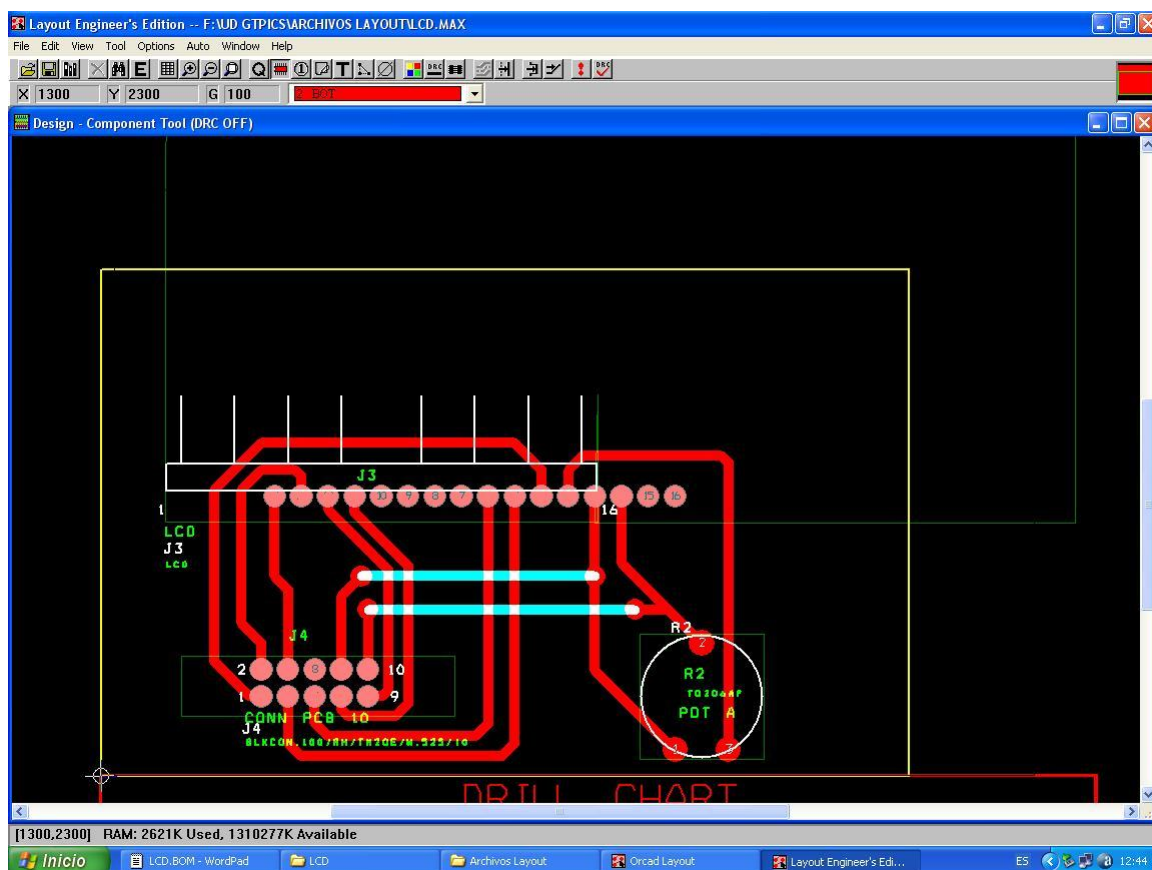
Comprobamos que tódolos compoñentes estean conxionados adecuadamente coa ferramenta “Design Rules Check”, creamos a lista de materiais necesarios coa ferramenta “Bill of Materials” e finalmente creamos o ficheiro necesario para o programa OrCad Layout coa ferramenta “Create Netlist”.

2.3.7 Deseño co programa OrCad Layout. Placa do LCD

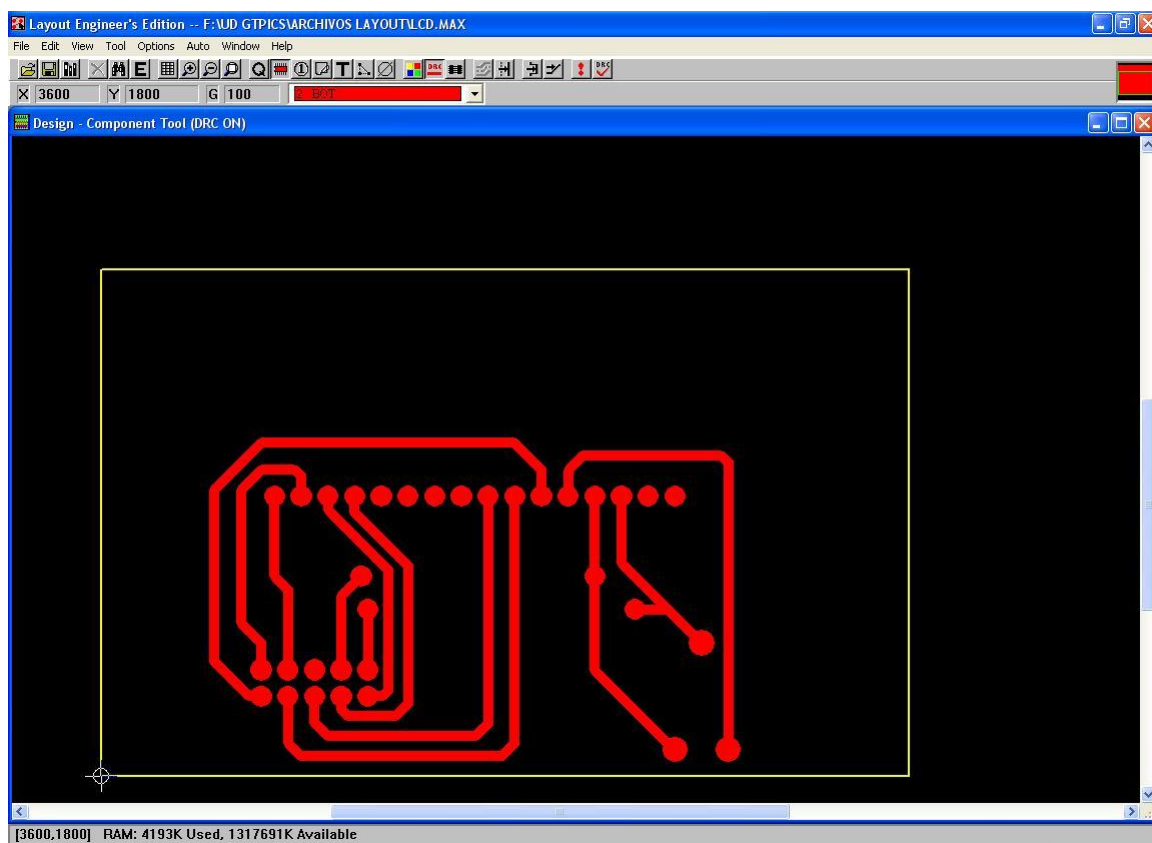
Abrimos o programa Orcad Layout e coa opción File → New seleccionamos o ficheiro de Tecnoloxía metric.tch. Engadimos o ficheiro de netlist creado co Orcad Capture anteriormente e seleccionamos a carpeta onde gardaremos o ficheiro Layout max file. Buscamos a mellor disposición dos compoñentes para o proceso de rotado da placa e obtemos unha disposición de compoñentes como podemos ver na figura seguinte:



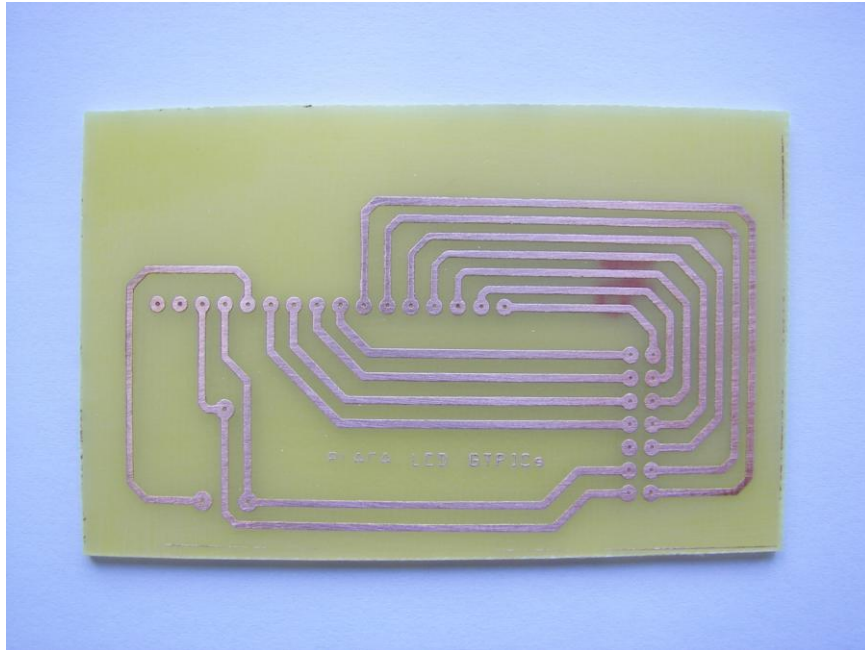
Procedemos ao proceso de rotado automático a simple cara da placa, co que obtemos o deseño de pistas da placa tal e como vemos na seguinte figura:



Sendo a cara de pistas:

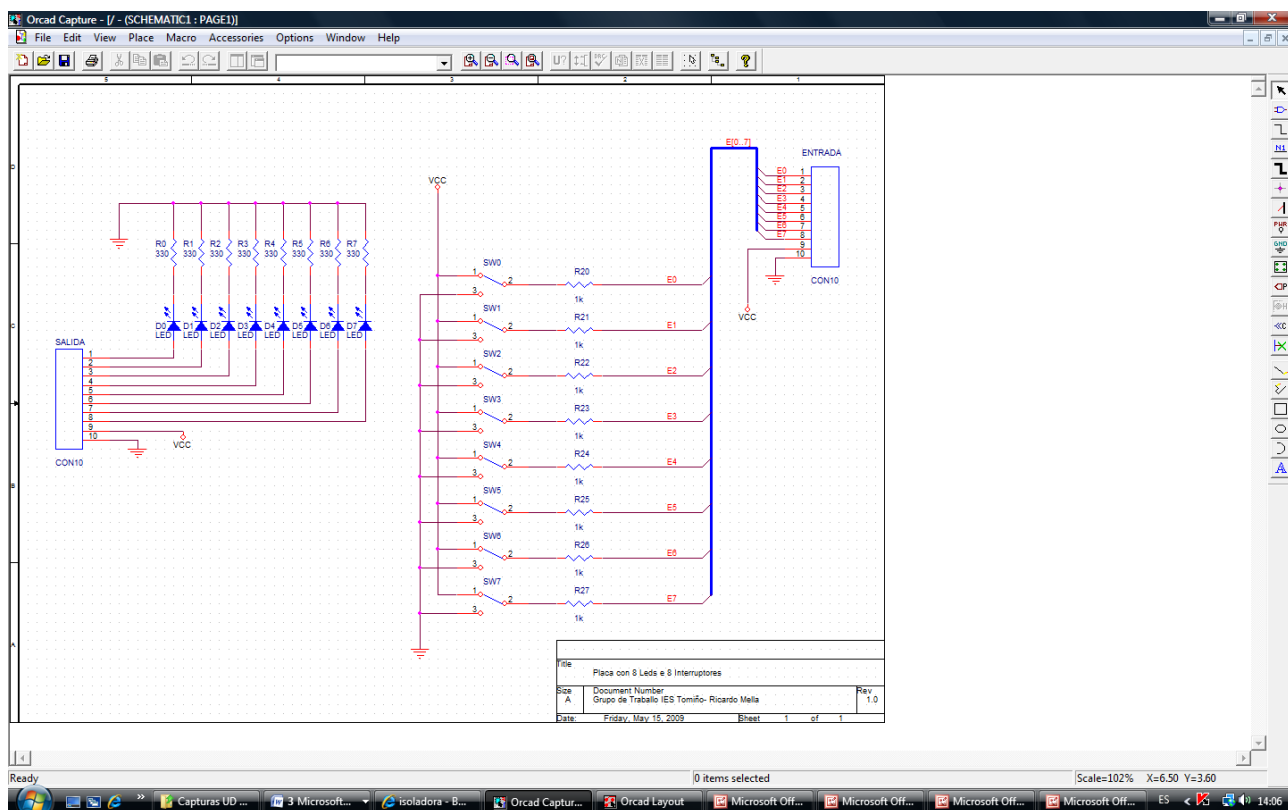


O seguinte proceso é a impresión destes esquemas en papel vexetal. Imos utilizar placa fotosensible de simple cara. Unha vez disposto o fotolito coa placa na insoladora, procedemos ao proceso de insolado axustando o tempo de exposición (podemos facer varias probas ata atopar o tempo correcto). Revelamos a placa e procedemos ao atacado obtendo un resultado como este:



2.3.8 Deseño co programa OrCad Capture. Placa dos LED e Interruptores

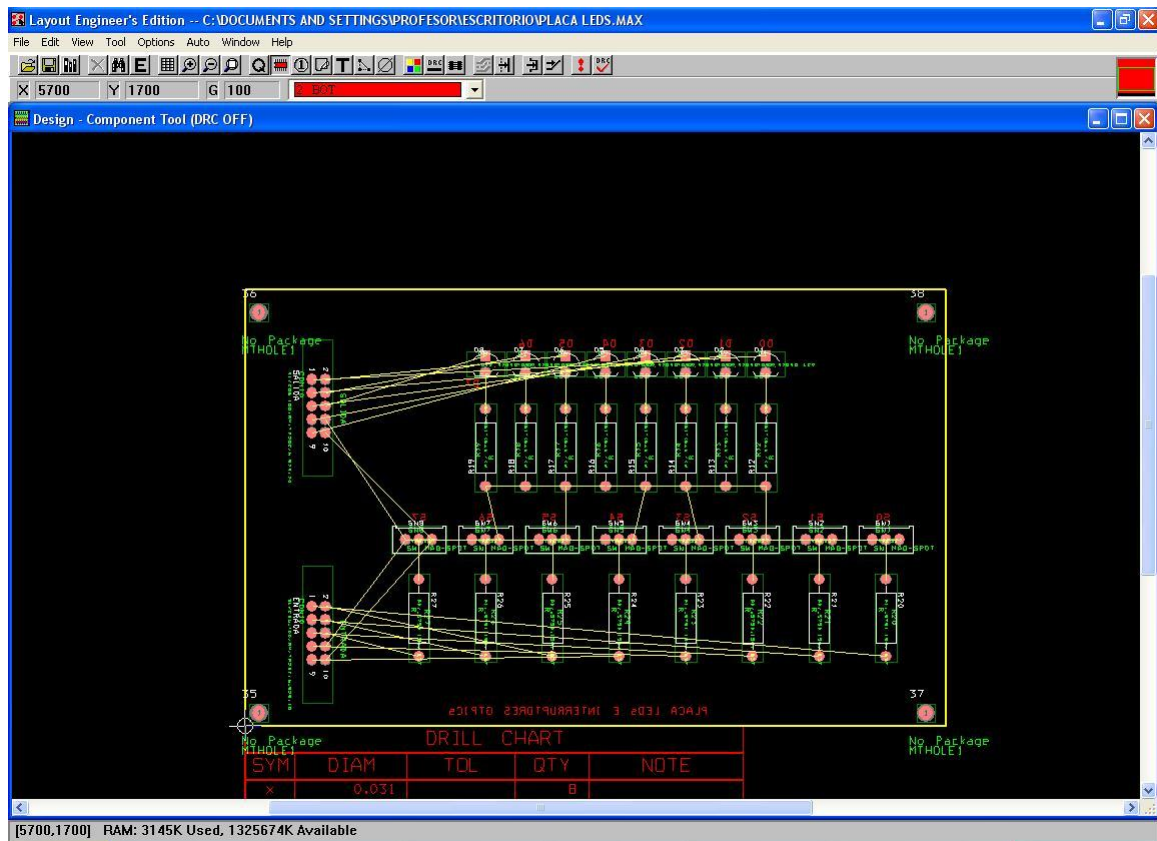
Abrimos o programa Capture e iniciamos un novo proxecto (seleccionamos o nome do proxecto e a carpeta onde o imos gardar) colocando todos e cada un dos distintos compoñentes que forman esta placa. No caso de que non atopemos algún deles nas librerías debemos facelo nos. Realizamos as conexións entre tódolos compoñentes e obtemos un esquema final da placa tal e como aparece na seguinte figura:



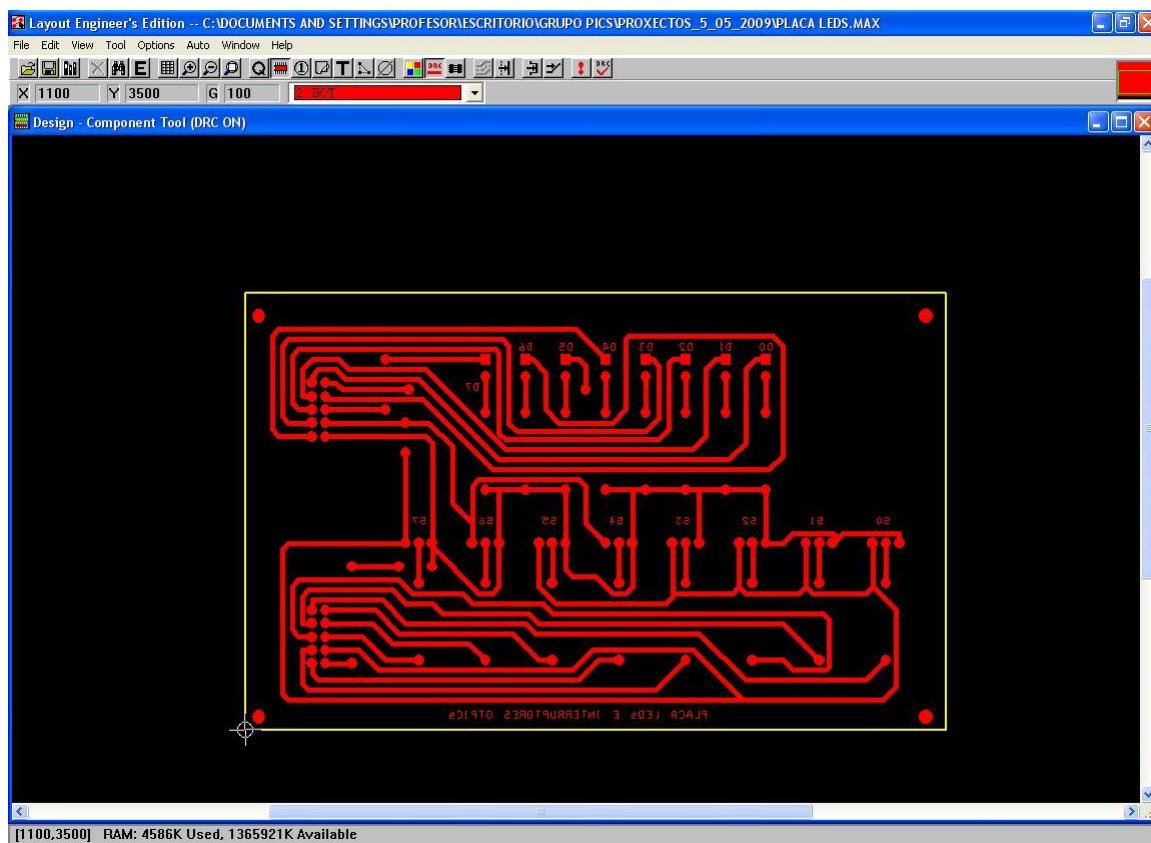
Comprobamos que tódolos compoñentes estean conxicionados adecuadamente coa ferramenta “Design Rules Check”, creamos a lista de materiais necesarios coa ferramenta “Bill of Materials” e finalmente creamos o ficheiro necesario para o programa OrCad Layout coa ferramenta “Create Netlist”.

2.3.9 Deseño co programa OrCad Layout. Placa dos LED e Interruptores

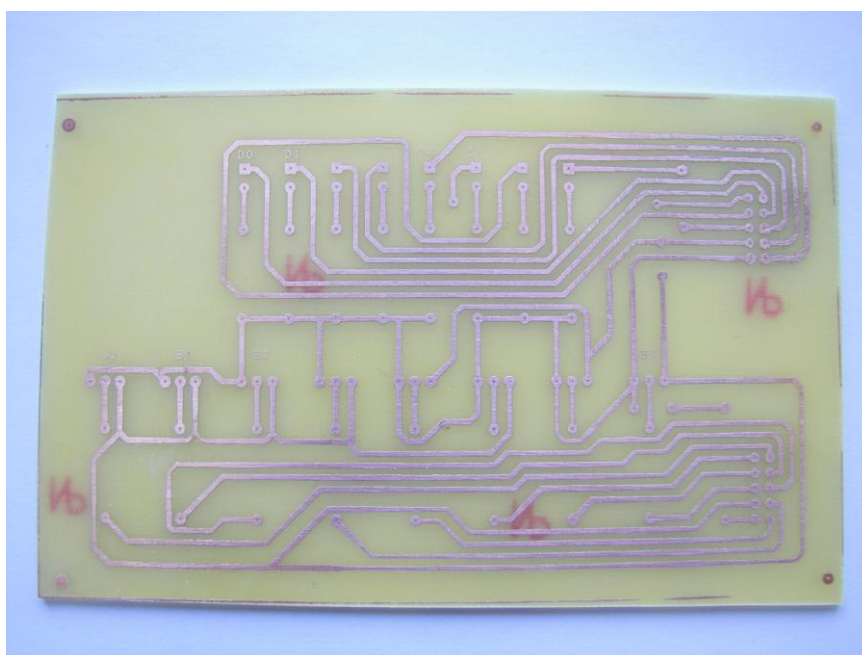
Abrimos o programa Orcad Layout e coa opción File → New seleccionamos o ficheiro de Tecnoloxía metric.tch. Engadimos o ficheiro de netlist creado co Orcad Capture anteriormente e seleccionamos a carpeta onde gardaremos o ficheiro Layout max file. Buscamos a mellor disposición dos compoñentes para o proceso de rotado da placa e obtemos o seguinte resultado:



Procedemos ao proceso de rotado automático a simple cara da placa, co que obtemos o seguinte deseño de pistas:



O seguinte proceso e a impresión destes esquemas en papel vexetal. Imos utilizar placa fotosensible de simple cara. Unha vez disposto o fotolito coa placa na insoladora, procedemos ao proceso de insolado axustando o tempo de exposición (podemos facer varias probas ata atopar o tempo correcto). Revelamos a placa e procedemos ao atacado obtendo un resultado como este:



Fonte de Alimentacion
 Entrada: Red 230V/50Hz
 Salidas: +5V, +3.3V, GND

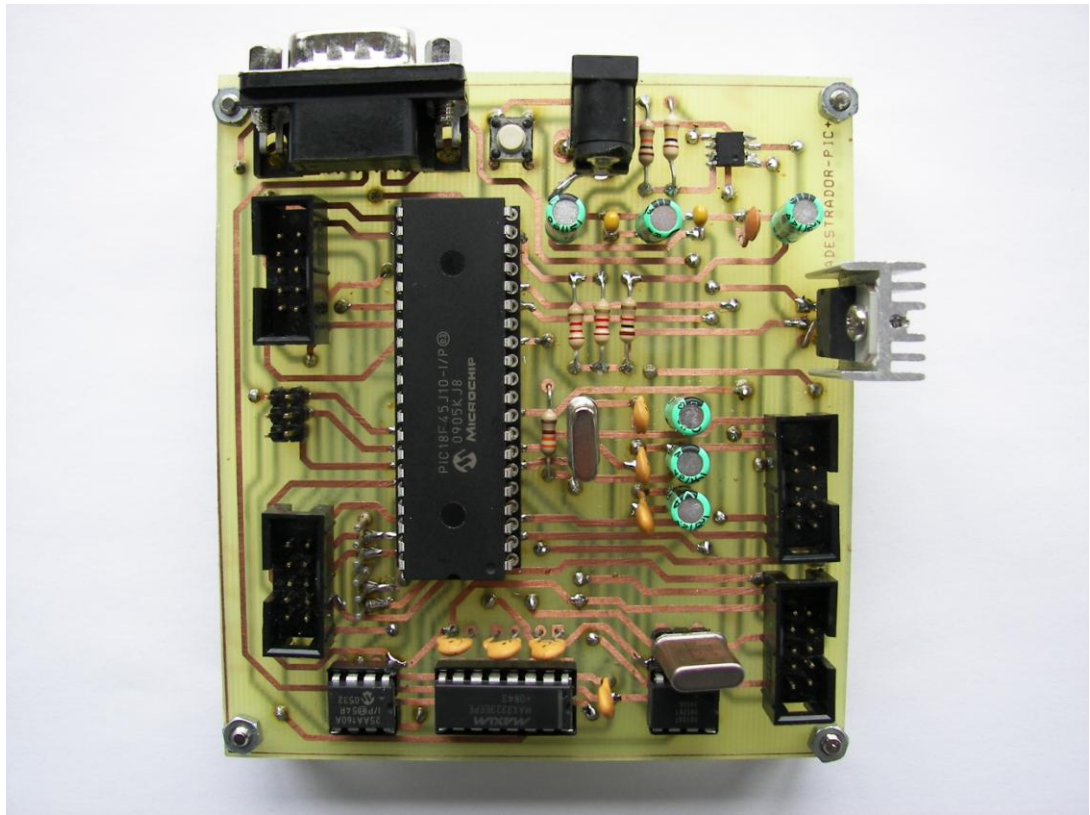
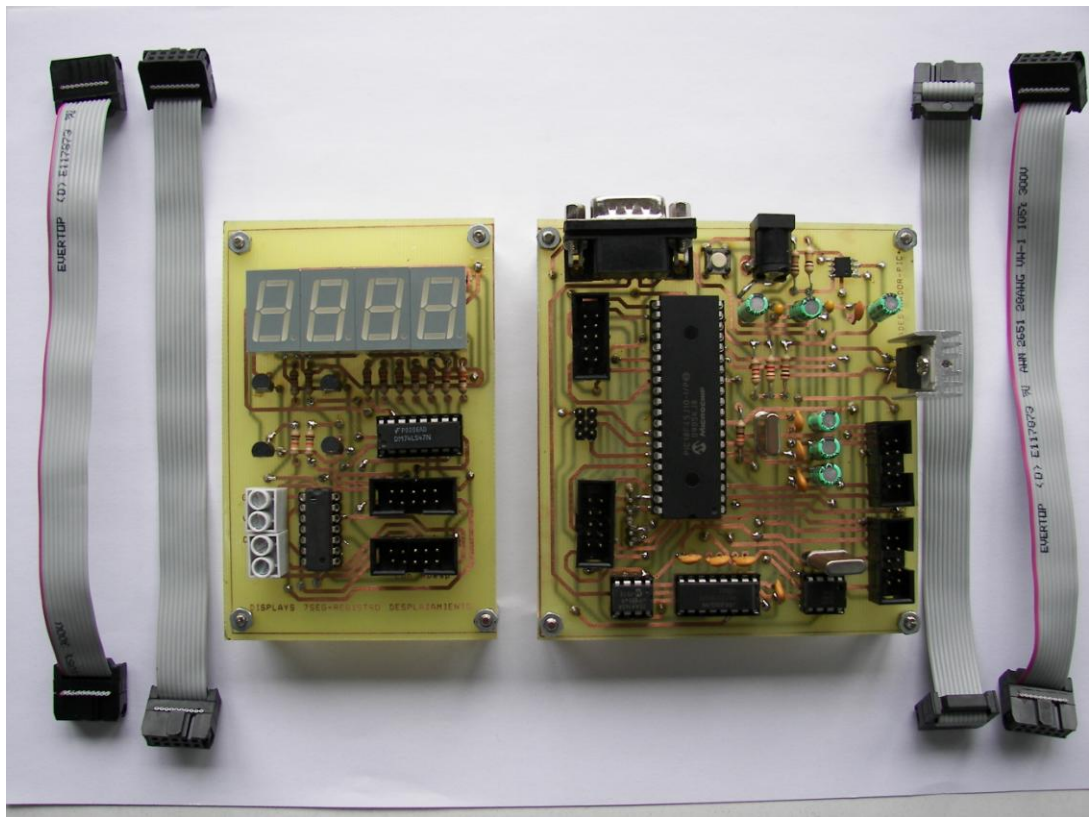
The schematic diagram illustrates a power supply circuit. It begins with a transformer (T1) connected to a red AC source (J1). The secondary winding of the transformer is connected to a bridge rectifier (B180C2300-1500B). The output of the rectifier is connected to a voltage doubler (U1: LM7805B/TO-220). The output of the voltage doubler is connected to a voltage regulator (U2: MCP1728-3302E-SN). The voltage regulator has three outputs: +5V, +3.3V, and GND. The +5V output is connected to a load (J2). The +3.3V output is connected to a load (J2). The GND output is connected to ground. The circuit includes various passive components such as resistors (R1, R2, R3), capacitors (C1, C2, C3, C4, C5), and diodes (D1, D2). The schematic is titled 'Fonte de Alimentacion'.

File	Fonte de Alimentacion: +5V, +3.3V, GND	
Size	Document Number	Rev
A	GRUPO DE TRABAJO IES DE TOMIÑO-RICARDO MELLA	01
Date:	Monday, May 25, 2009	Sheet 1 of 1

2.3.10 Imaxes do Programador do PIC, Placa PIC, Displays e Rexistros de Desprazamento e conectores utilizados para comunicar as placas

O programador que utilizaremos para programar o adestrador deseñado por nos é o “PICkit 2” da casa Microchip. Podemos velo na seguinte figura:







2.3.11 Anexo

Engadimos como anexo os esquemas da Placa do PIC, Placa de LED e Interruptores, Placa LCD e Displays 7 segmentos con decodificador e rexistro de desprazamento e Fonte de alimentación. Engadimos tamén os fotolitos (excepto da fonte de alimentación xa que nos a incluímos na placa do PIC) e unha lista de compoñentes de tódolos esquemas anteriores.