Construya su propio decodificador Quad Servo DCC - Actualización

Introducción

Siguiendo una serie de comentarios de los aficionados que construyeron Decodificadores DCC Quad Servo (QSDD), tal y como fue publicado en los números de **febrero** y **marzo** de 2020 de la revista MRH, ahora he realizado varios mejoras tanto en el software como en el hardware de la versión original.

Manejo de múltiples comandos de servo

La versión publicada del software (4.6) sólo permitía que se moviera uno de los cuatro servos adjuntos en cualquier momento, lo cual era un poco limitante al establecer rutas que implicaban una serie de desvíos. Así que me puse a agregar un código para permitir el funcionamiento simultáneo de todos los servos.
Sólo para descubrir que Geoff Bunza, que escribió el código original en el que se basa el QSDD, había incorporado ya esta funcionalidad. El problema radicaba en la forma incorrecta en que había modificado y luego llamado a las rutinas de Geoff. Ahora he arreglado esto para que todos los servos puedan responder simultáneamente a Comandos DCC.

Programable a través de JMRI Decoder Pro

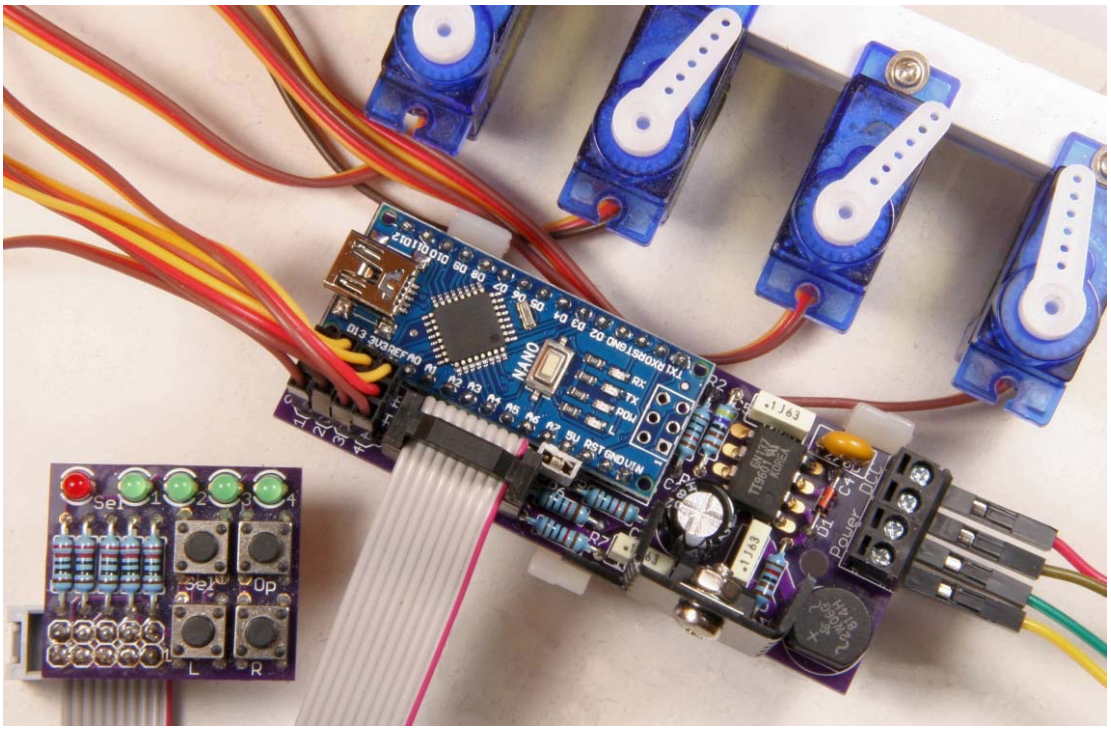
Mientras investigaba el problema de múltiples comandos, me contactó otro constructor de QSDD, Drew Aldridge de California, que deseaba poder acceder a las variables de configuración de QSDD mediante el Decodificador JMRI Pro. En mi artículo original, sugerí que la razón por la que esto falló fue que JMRI no reconocía que los decodificadores de accesorios podían tener CV, ya que en la mayoría de los accesorios comerciales los decodificadores no los usan. Sin embargo, Drew señaló que podía leer y escribir CV de otros decodificadores accesorios que los usaban, así que esta vez investigué adecuadamente. El verdadero problema radicaba en la inicialización de mi QSDD, que tardaba demasiado en completarse después aplicarse a la vía de programación. Debido a que Decoder Pro no estaba obteniendo una respuesta oportuna, legítimamente cicló energía a la pista y al QSDD, que simplemente repitió el ciclo de falla ***ad infinitum***...

La solución, ahora implementada, era acelerar la inicialización de QSDD, que ahora requiere un máximo de 0,15 segundos. Tan pronto como se aplica energía, en lugar de cambiar tranquilamente todos los LED del teclado QSDD encendidos uno por uno, y luego apagándolos nuevamente, ahora verá solo un breve parpadeo del LED rojo de selección antes de que el QSDD esté listo para JMRI Decoder Pro en su pista de programación, lo que le permite leer y escribir todos los CV de QSDD (como se enumeran en la Parte 2 del artículo original, o en la documentación en mi sitio web en (**https://www.a-train-systems.co.uk/download.htm#Projects**). Al agregar el QSDD a su lista en Decoder Pro, haciendo clic en New Loco, seleccione NMRA como el fabricante (en la parte superior de la lista) y “NMRA accessory decoders” como el tipo de decodificador.
El software actualizado para Arduino Nano (QuadServo_DCC-Decoder_5-2.ino) también se puede descargar de la página de mi sitio web, como se indica arriba.

Consulte las notas al final de este documento para obtener detalles sobre cómo acceder a las variables de configuración.

Alimentación externa

Varios usuarios no estaban interesados en que el QSDD recibiese toda la alimentación, incluidos los servos adjuntos, del suministro DCC de la vía, ya que la corriente de bloqueo tomada por un servo puede ser superior a 600 mA. Por lo tanto se ha realizado un cambio menor en el hardware QSDD para proporcionar una entrada de energía que aceptará cualquier suministro de 9 a 15 voltios CA o CC. La entrada DCC del la pista entonces sólo tiene que suministrar los 10mA aproximadamente, tomados por el optoacoplador. Los detalles del cambio se puede encontrar en el diagrama esquemático actualizado al final de este documento.
Dado que no había una forma sencilla de modificar la placa de circuito impreso existente para separar el DCC de la entrada de la fuente de alimentación QSDD, se diseñó una nueva PCB con un par de terminales adicionales para la fuente de alimentación externa, como se muestra en la siguiente imagen. Si está conforme con alimentar todo del bus desde la vía DCC, simplemente conecte ambos conjuntos de terminales a la alimentación DCC.



Todas las fotografías del autor

Como se puede ver en la fotografía, el QSDD sigue usando exactamente el mismo pequeño teclado desmontable que se conecta al decodificador y se utiliza para ajustar manualmente los ángulos del servo y configurar su velocidad de movimiento.

Como antes, el teclado también se puede usar para operar los servos a mano, usando la botonera. Para mayor comodidad, además de poder conectarse directamente al decodificador, el teclado se puede conectar alternativamente a través de un cable plano de hasta 40 pulgadas de largo, lo que le permite ver y configurar sus desvíos cuando el decodificador y los servos estén fuera de la vista, debajo de la maqueta.

Una vez configurado, no es necesario mantener el teclado enchufado. Si lo desea, puede desconectarlo completamente de la placa decodificadora. El decodificador luego moverá los servos conectados para operar sus desvíos en respuesta a los comandos estándar de DCC enviados a la vía desde su central, utilizando controladores de mano o software adecuado (como **JMRI Panel Pro** o el mío Aplicación **A-Track**) ejecutándose en su ordenador.

Nueva PCB disponible

Para admitir el uso de una fuente de alimentación externa, se incluye una placa de circuito impreso revisada para el QSDD, y disponible en OSH Park en (**https://oshpark.com/shared_projects/zMo5jegR**) donde puede ordenar un conjunto de PCB, o descargar el archivo de la placa (en formato EAGLE, .brd) para enviarlo a cualquier otro fabricante de PCB de su elección.

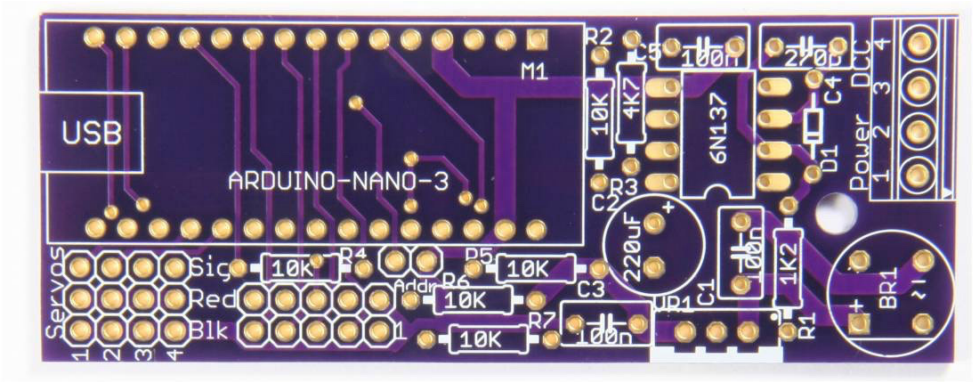
La PCB QSDD-2 es 0.1” más larga que la versión original, por lo que cuesta unos centavos más, pero, como bono adicional, ahora incluye un solo agujero de diámetro de 3,2 mm. (junto al bloque de terminales y el puente rectificador) para ayudar a montar la unidad ensamblada donde sea necesario. El QuadServo-PCB del teclado también se puede pedir a OSH Park si es necesario (**https://oshpark.com/shared_projects/7ATX5aqB**).

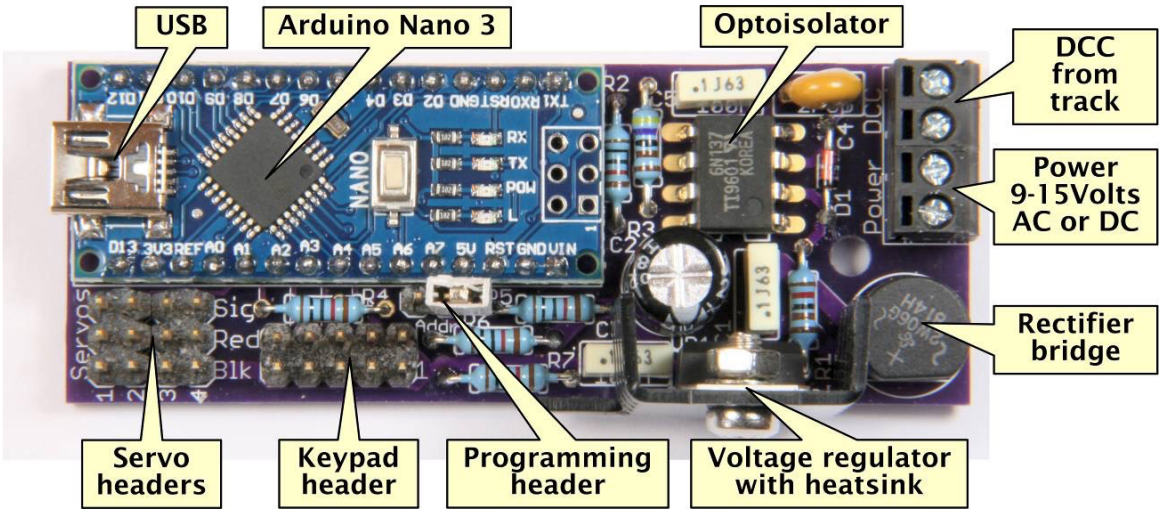
La lista de piezas y los detalles de montaje del QSDD (como se publicó en la Parte 1 del artículo original) permanecen esencialmente como antes, con la única adición de un segundo bloque de terminales de 2 vías para la alimentación externa independiente.

Ambas versiones de QSDD ejecutan el mismo software (versión actualizada 5.2) y la configuración manual.
Los procedimientos y la operación a través del teclado siguen siendo exactamente los mismos que se describen en el artículo original (y en el video MRH [1])adjunto en https://www.youtube.com/watch?v=Ox-X1uAWssc&feature=youtu.be).

Detalles de montaje

Los únicos cambios en la PCB QSDD-2, después de la adición del bloque de terminales de alimentación y el orificio de montaje, están en la ubicación de los componentes alrededor del optoacoplador y el regulador de voltaje, como se puede ver si las fotografías a continuación se comparan con las de la Parte 1 del artículo original publicado.



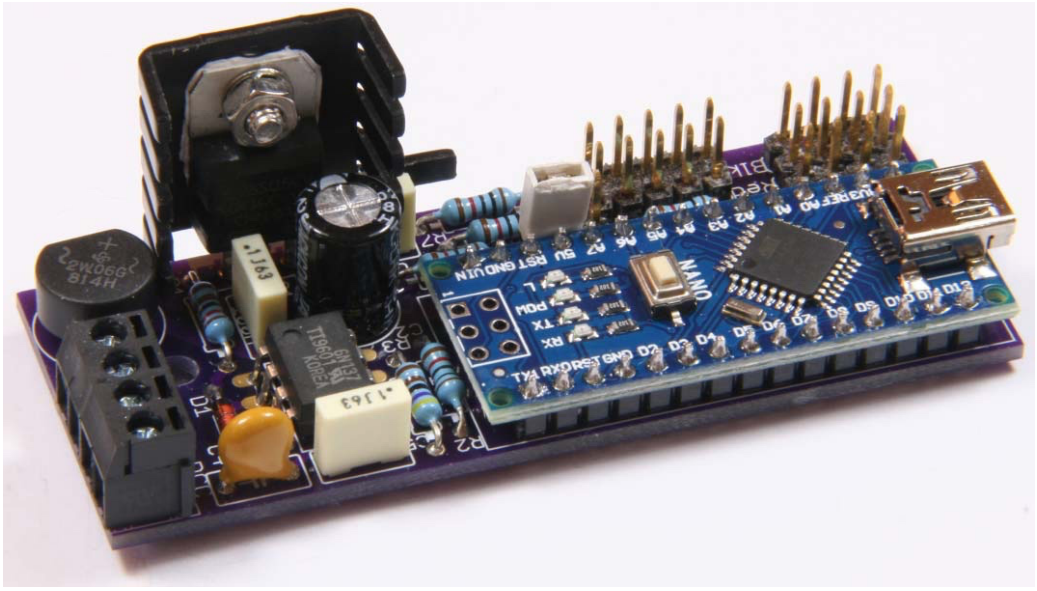


Consulte también la Parte 1 del artículo original para obtener una lista completa de piezas junto con las fuentes de suministro sugeridas. Tenga en cuenta que se instalan dos bloques de terminales de 2 vías en lugar de un solo elemento de 4 vías, ya que generalmente es una opción más barata. Encontrará que los bloques de terminales de 2 y 3 vías cuestan menos, por terminal (especialmente si se compran a granel), que los bloques más grandes y, en la mayoría de los fabricantes, los bloques pequeños están diseñados para engancharse cuidadosamente para formar bloques más grandes con cualquier número requerido de terminales.

El montaje del QSDD-2 actualizado, más un teclado si es necesario, sigue exactamente los mismos pasos que el original con respecto al orden de montaje de los componentes. Una pequeña diferencia en la etapa final de montaje es que, si el disipador de calor identificado en la lista de piezas está equipado con las aletas hacia adentro –hacia el centro de la PCB–, solo necesita enderezar con cuidado uno de los pares inferiores de aletas (tenga cuidado con el aluminio blando que se agrieta fácilmente y separe la aleta del cuerpo principal del disipador de calor), para librar el condensador C3.

1.- Aunque el vídeo está en inglés, visualmente es lo suficientemente claro en la exposición del funcionanmiento del decodificador.

La otra aleta inferior ahora encaja en el espacio entre el puente rectificador y la resistencia R1, como se muestra en la fotografía siguiente:



Con el QSDD-2 ensamblado, ahora puede conectar el teclado al cabezal de 2 x 5 pines en el decodificador, ya sea directamente o utilizando el cable plano descrito en la Parte 1, y conectar el cable USB (conector A a conector Mini B) de su ordenador al módulo Arduino Nano. Si todo es ensamblado correctamente, el Arduino Nano debe encender dos de sus LED integrados, uno fijo y otro intermitente (dependiendo del estado interno actual del Nano), listo para recibir su programa de software (**QuadServo_DCC-Decoder_5-2.ino**).

Detalles completos de cómo configurar el IDE de Arduino en su ordenador y luego usarlo para descargar el software para QSDD se pueden encontrar en la Parte 1 del artículo original, junto con los pasos iniciales necesarios para configurar y probar su decodificador recién ensamblado.

Los pasos adicionales necesarios para programar direcciones de servo de su propia elección en el decodificador, y también para configurar los movimientos de servo con precisión para adaptarse a los enlaces al desvío en su maqueta, se cubrieron en la Parte 2 del artículo original, y siguen siendo exactamente iguales para la versión actualizada.

Acceso a variables de configuración y operaciones DCC

Puede acceder a las Variables de configuración (CV) de QSDD directamente utilizando el controlador de mano y siguiendo las instrucciones del fabricante, o a través de su ordenador (y la Central de su sistema) usando JMRI Decoder Pro o mi propia aplicación A-Track.

Sin embargo, en todos los casos deberá conectar su QSDD a la Vía de Programación. Mientras esto era sencillo con el hardware QSDD original (con una sola conexión DCC), con el hardware QSDD-2 actualizado, debe desconectar la fuente de alimentación externa (y cualquier Cable USB) del decodificador y conectar ambos conjuntos de terminales (DCC y alimentación externa) a la Vía de Programación. Esto asegura que la estación de comando reciba la retroalimentación adecuada del QSDD y su teclado adjunto, al leer o escribir CV, en forma de una pequeña sobretensión cuando todos los LED del teclado se encienden brevemente.

El conjunto de variables de configuración utilizadas por el QSDD para mantener sus datos de trabajo permanece más o menos como se presenta en la Parte 2 del artículo original, con la adición de una Versión de Decodificador Extendido (CV109 - CV112) para identificar el decodificador como QSDD junto con la versión de software, tal y como están enumerados en la tabla siguiente:

CV No.	Default Value	Description
01	1	Board Address LSB – internal use – ignore any value loaded here
07	89	NmraDCC Version
08	13	Manufacturer (Do-It-Yourself)
09	0	Board Address MSB – internal use – ignore any value loaded here
29	226	Decoder Configuration – Extended Accessory + Output Addressing
41	1	Output 1 Address LSB
42	0	Output 1 Address MSB
43	2	Output 2 Address LSB
44	0	Output 2 Address MSB
45	3	Output 3 Address LSB
46	0	Output 3 Address MSB
47	4	Output 4 Address LSB
48	0	Output 4 Address MSB
50	0	Load Default CV Values if Not = 173 (0xAD), Auto set = 173 after load
55	6	Servo 1 - Rate (1 = Fast to 16 = Slow)
56	1	Direction - 1 = Normal Operation, 0 = Reverse Operation
57	70	Right Limit
58	110	Left Limit
59	70	Current Position
60	6	Servo 2 - Rate (1 = Fast to 16 = Slow)
61	1	Direction - 1 = Normal Operation, 0 = Reverse Operation
62	70	Right Limit
63	110	Left Limit
64	70	Current Position
65	6	Servo 3 - Rate (1 = Fast to 16 = Slow)
66	1	Direction - 1 = Normal Operation, 0 = Reverse Operation
67	70	Right Limit
68	110	Left Limit
69	70	Current Position
70	6	Servo 4 - Rate (1 = Fast to 16 = Slow)
71	1	Direction - 1 = Normal Operation, 0 = Reverse Operation
72	70	Right Limit
73	110	Left Limit
74	70	Current Position
109	81	Extended Decoder Version 1 – "Q"
110	83	Extended Decoder Version 2 – "S"
111	68	Extended Decoder Version 3 – "D"
112	82	Software Version – Hex = 0x52

Además de este “conjunto de trabajo”, hay otras tres CV que controlan funciones “**especiales**”:

CV No.	Default Value	Description
120	0	Set = 120 to load Default CVs – must be cleared manually
121	24	Address to set CV Values in Operations Mode (Program on Main) – LSB
122	0	Address to set CV Values in Operations Mode (Program on Main) – MSB

Veamos primero las funciones “especiales”: la dirección contenida en las CV 121 y 122 se utiliza para escribir un nuevo valor a cualquier CV dentro del decodificador usando el Modo de Operaciones (también llamado Programación en el Principal). Configure su sistema DCC a través de su controlador de mano en modo de operaciones y prepárese para utilizar una dirección de locomotora (no una dirección de accesorio) igual a la dirección contenida en CV 121 y 122, que es 24 por defecto.

Seleccione el número de CV que se programará, introduzca el nuevo valor y presione ENTER (o la tecla destinada por su sistema para completar el comando). Dado que no es posible volver a leer valores de CV en el modo de operaciones, tendrá que juzgar, por el comportamiento posterior del decodificador, si el cambio de valor de CV fue un éxito.

La otra función “especial”, iniciada programando un valor de 120 en CV120 (a través del Modo Operaciones), es manejado por la biblioteca NmraDcc y carga todos las CV con sus valores predeterminados la siguiente vez que el decodificador se reinicia o se enciende. El restablecimiento se logra presionando el botón Reset en el Arduino Nano, o manteniendo presionados los botones **L** y **Op** en el teclado y liberándolos seguidamente.

Sin embargo, parece que la CV120 no es borrada automáticamente por la biblioteca NmraDcc, por lo que la CV continuará restableciéndose a sus valores predeterminados cada vez que se reinicie el decodificador hasta que cambie el valor en CV120 usted mismo.

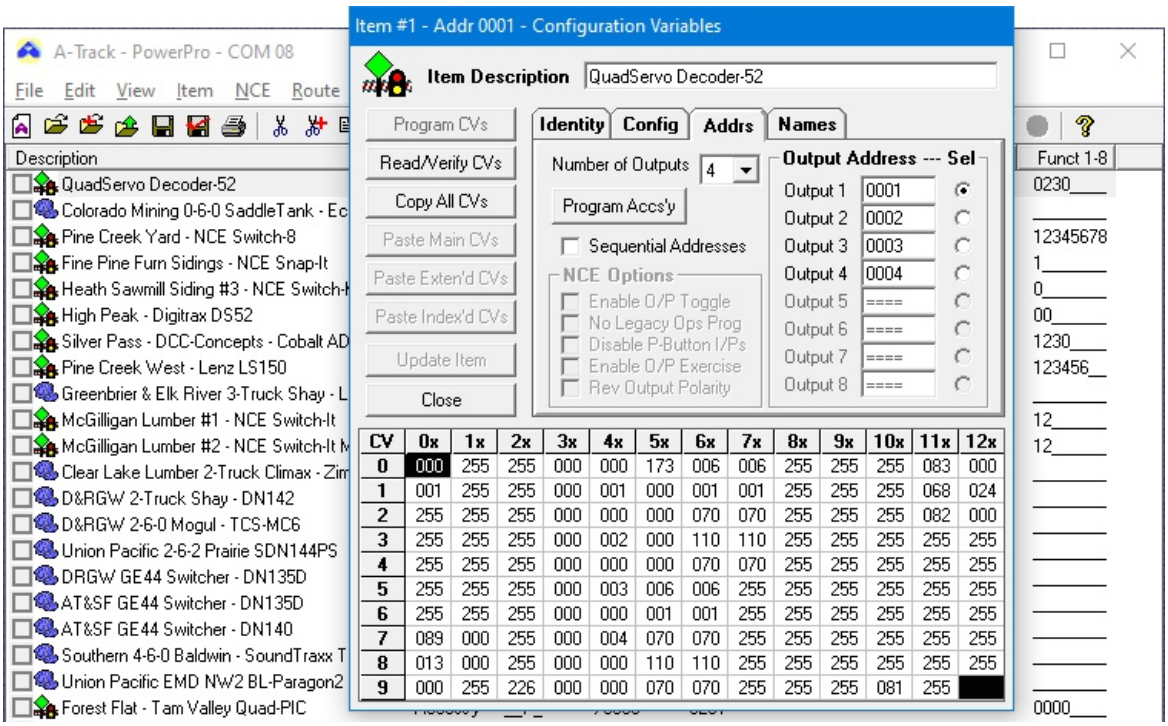
Para evitar este inconveniente, el QSDD se ha configurado para utilizar la CV50 como alternativa. Cualquier valor *excepto 173* en esta CV restablecerá todas las CV a sus valores predeterminados la próxima vez que encienda o reinicie el decodificador, después de lo cual el valor de CV50 se establecerá en 173 automáticamente (hex 0xAD = “All Default”). Esto significa que la carga predeterminada solo ocurre una vez, y ocurrirá automáticamente cuando el sketch del decodificador se cargue en el Arduino Nano por primera vez (cuando todas las CV contendrán normalmente el valor 255).

Suponiendo que tiene el QSDD conectado a su Vía de Programación, como se describe en la introducción a esta sección del documento, y tiene el teclado adjunto al decodificador, a continuación puede leer o escribir cualquiera de los valores de CV. Después de la actualización de software a la versión 5.2 todas las restricciones sobre el tipo de sistema DCC que puede utilizar para la programación, como se describe en la Parte 2 del artículo original ya no se aplica, y cualquier versión de Command Station funcionará.

Si tiene algún tipo de sistema NCE DCC más una ordenador con Windows, mi propia aplicación, A-Track (**www.a-train-systems.co.uk/atrack**), leerá y programará con gusto los CV de QSDD y le permitirá para guardar un registro de ellos en el archivo. Puede descargar A-Track y su documentación completa desde mi sitio web (**https://www.a-train-systems.co.uk/download.htm#ATWindows**).

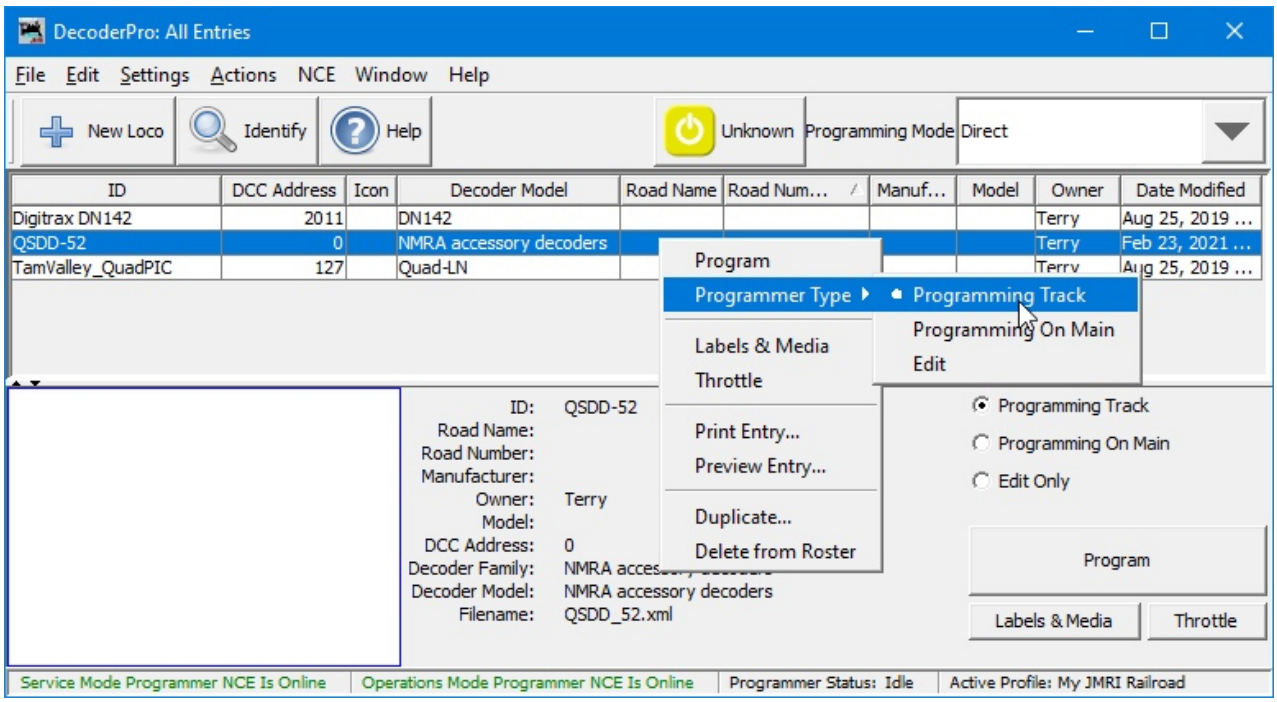
La principal ventaja de poder guardar un conjunto completo de CV en un archivo se muestra cuando se tiene una maqueta con muchos desvíos y múltiples QSDD para accionarlos. Después de configurar un decodificador, y sus cuatro desvíos asociados, a su entera satisfacción, puede tomar una copia de las CV modificadas y transferirla en una sola operación a todos sus otros QSDD.
Configurar estos otros QSDD, cuando se transfieren de la vía de programación a la maqueta, supondrá solo pequeños ajustes a los desplazamientos del servo para compensar las diferencias entre servos individuales (y quizás desvíos o enlaces).

La siguiente captura de pantalla de la aplicación A-Track muestra la ventana Variables de Configuración para el decodificador con todas las CV relevantes configuradas en sus valores predeterminados.

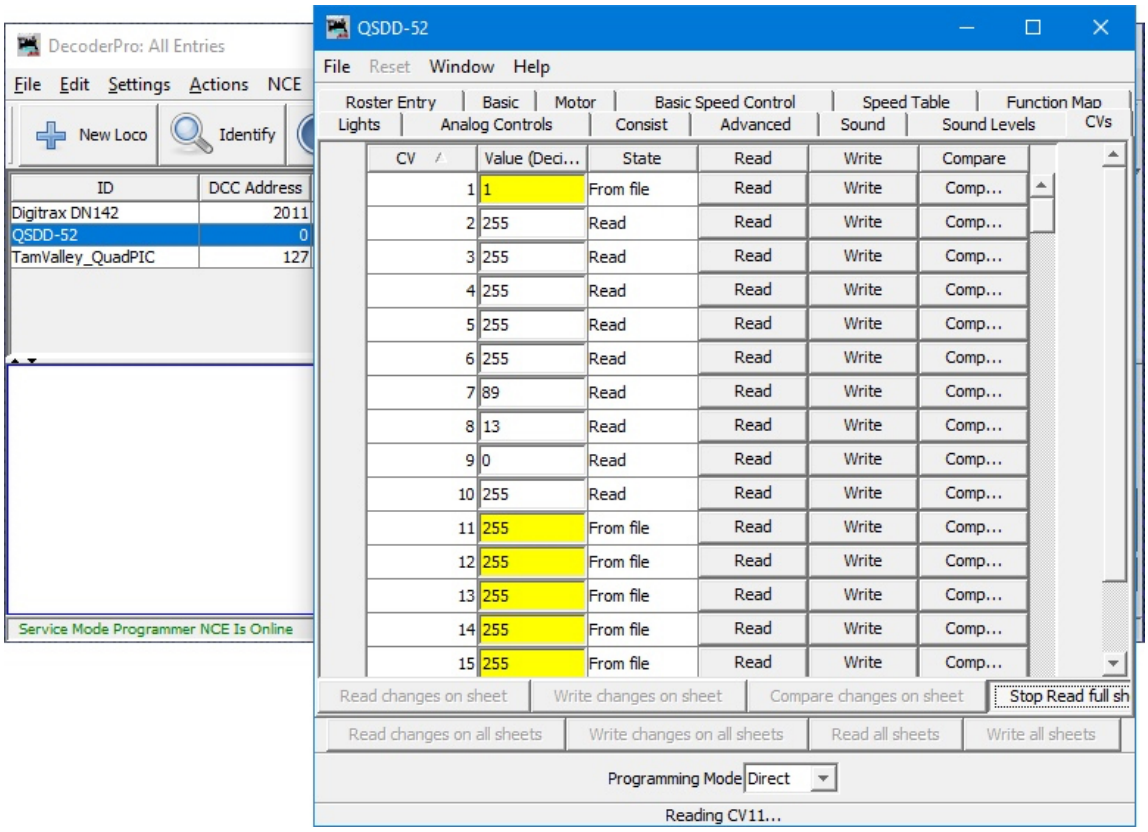


Los ajustes de cualquiera de los valores de CV se pueden hacer simplemente escribiendo un nuevo valor (o conjunto de valores y luego usar la función de CV del programa para transferirlos al decodificador.

Con centrales distintas de NCE, puede utilizar el software Decoder Pro de JMRI (**https://www.jmri.org/help/en/html/apps/DecoderPro/index.shtml**) para leer y programar de manera similar CV QSDD y para guardar una copia en un archivo en su ordenador. Agregar el QSDD a su lista en Decoder Pro se realiza haciendo clic en New Loco, seleccionando NMRA como fabricante (en la parte superior de la lista) y luego “NMRA accessory decoders” como tipo de decodificador.



A continuación, puede acceder al QSDD en la Vía de Programación, y luego hacer clic en "**Program**" para abrir la ventana de programación; seleccione la pestaña 'CV' y haga clic en "**Read full sheet**" para comenzar a leer las variables de configuración de QSDD:



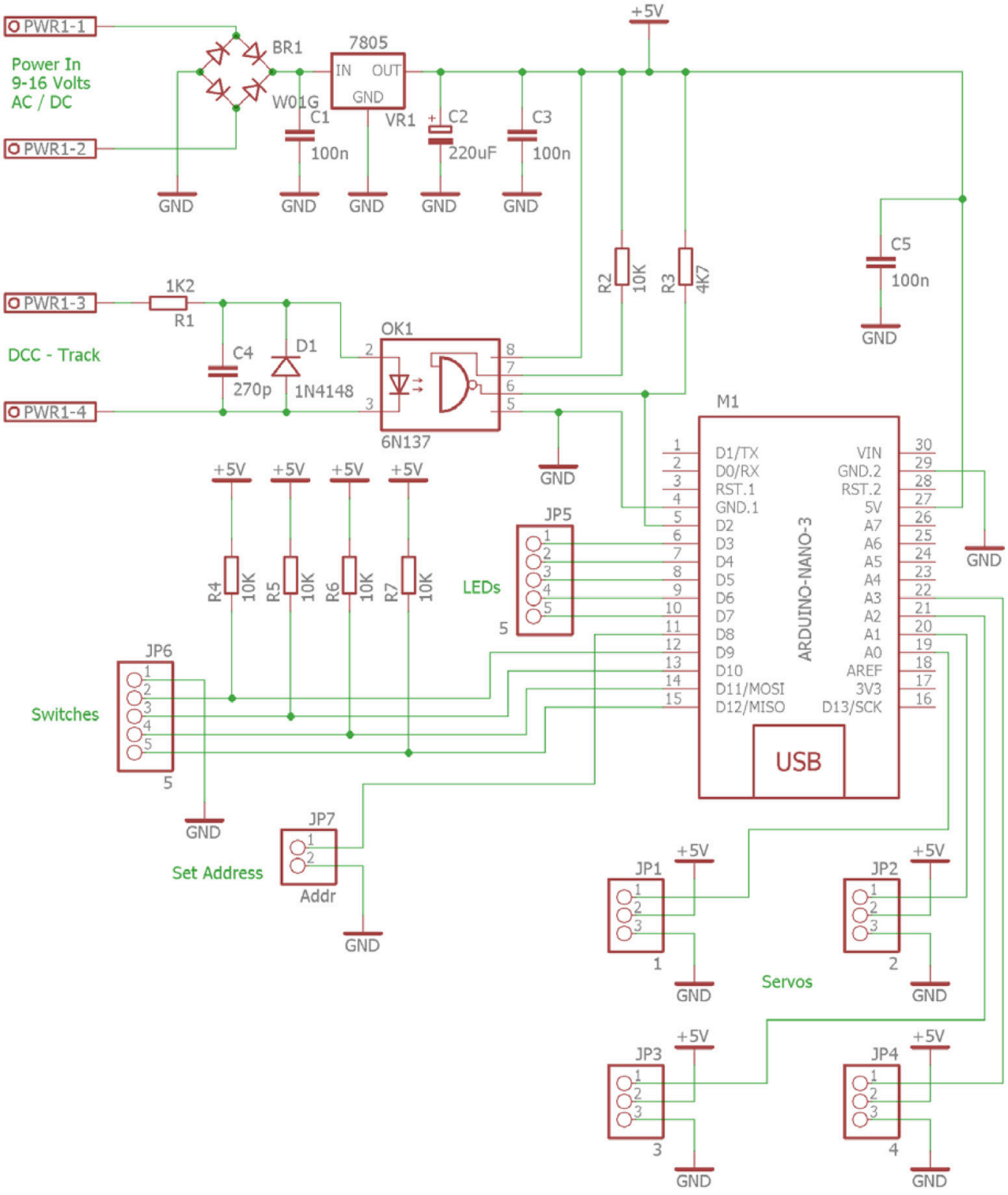
Detalles técnicos

Para quién que esté interesado, el esquema del circuito de la placa decodificadora QSDD-2 actualizada es la que se muestra en la página siguiente. El esquema de la placa del teclado no ha cambiado respecto al presentado en la Parte 2 del artículo original.

La energía para la placa de una fuente externa (9 - 15 voltios CA o CC) se rectifica según sea necesario por el puente de diodos BR1 para alimentar el regulador de voltaje VR1, que a su vez suministra +5 voltios al Arduino Nano, sus circuitos asociados y los servos adjuntos. Aunque la corriente normal a través del regulador, con los servos inactivos, es de unos 45 mA, mantiene su disipación de potencia por debajo de 0,5 W, y podría aumentar brevemente hasta 10 W si se aplica el voltaje de entrada máximo y se ordena el funcionamiento simultáneo de los cuatro servos. De ahí la necesidad del disipador de calor adjunto al regulador.

El circuito de entrada DCC está adaptado del diseño de Wolfgang Kuffer (https://mrrwa.org/dcc-decoder-interface/) y es utilizado por Geoff Bunza como base para varios de sus proyectos.
La señal de entrada DCC se conecta a la entrada del optoacoplador OK1 a través de la resistencia R1. El condensador C4 filtra cualquier pico de voltaje de la vía, y el diodo D1 evita que la entrada del diodo del optoacoplador sea fatalmente polarizado inversamente por la parte negativa de la señal DCC.

La salida del optoacoplador OK1 es una réplica de la forma de onda DCC, pero a un nivel seguro de +5 voltios, por lo que los paquetes de comando DCC se pueden introducir a la entrada digital D2 del módulo Arduino Nano. Aquí son decodificados por las funciones de la biblioteca NmraDcc, y los comandos de accesorios relevantes, y luego pasado al código del boceto QSDD.



Decodificador Quad Servo DCC - Decodificador

El resto del circuito cubre las diversas entradas y salidas del Arduino Nano.

Los LED montados en el teclado se controlan desde las salidas D3 - D7 a través del conector JP5 en el decodificador y el conector JP1 en el teclado con resistencias R1 - R5 configurando la corriente a través de cada LED a, aproximadamente, 13 mA.

Los pines D9 - D12 se establecen como entradas que se conectan a los cuatro botones pulsadores montados en el teclado mediante el conector JP6 del decodificador y el conector JP2 del teclado.

Las entradas están normalmente a un nivel ALTO por las resistencias R4 - R7 en el decodificador, pero se llevan a un nivel BAJO (GND ó 0 voltios) siempre que se presione el botón correspondiente.

El pin D8 se lleva de manera similar a un nivel BAJO siempre que el puente que cortocircuita los dos pines de programación JP7, está instalado. Este pin utiliza un pull-up interno en lugar de una resistencia externa adicional para que mantenga su nivel ALTO cuando el puente no esté instalado.

Finalmente, los pines A0 - A3 están configurados como salidas para impulsar uno de los cuatro servos conectados a través de los cabezales de tres pines, JP1 - JP4 en la placa decodificadora.