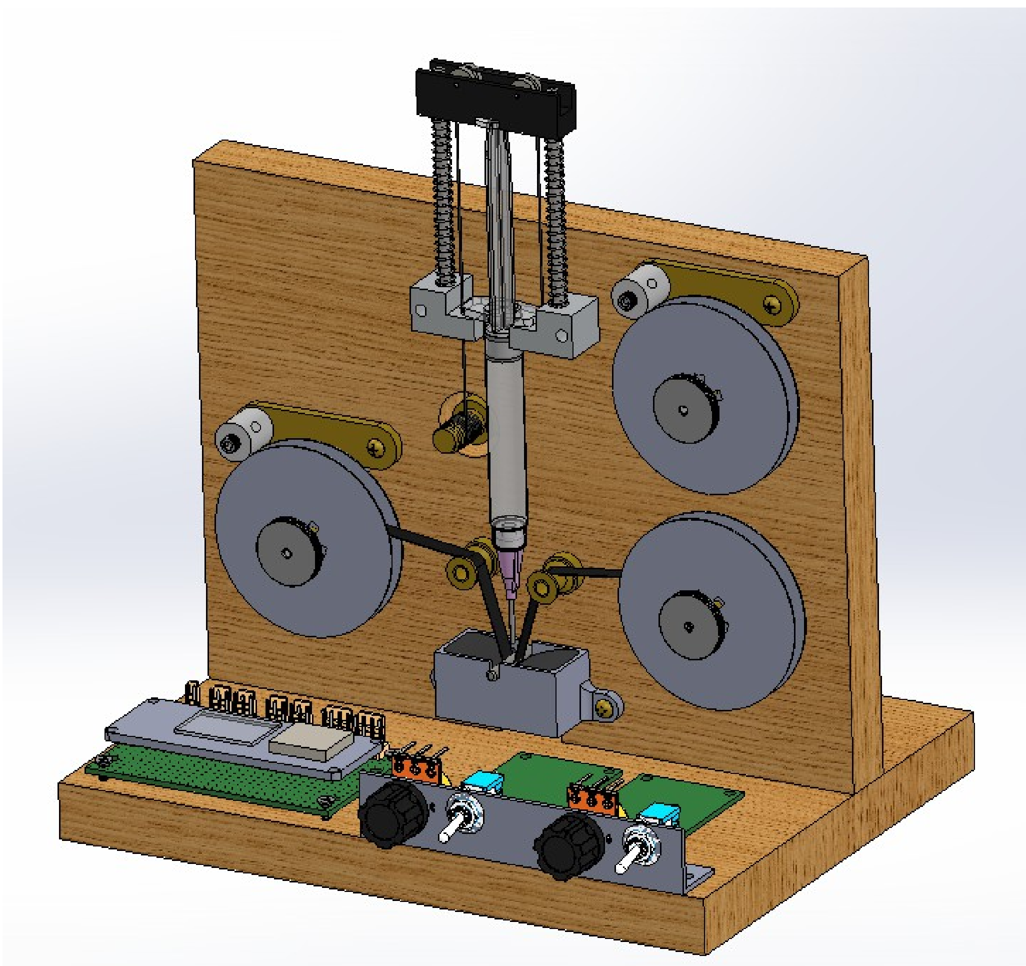
IBM 029 Keypunch Ribbon Ink Applicator

User Manual & Report



Document Revision History

Version	Author	Date	Document Release / Revision Description
1	DC	23 Feb., 2023	Initial Release

Table of Contents

1 Introduction

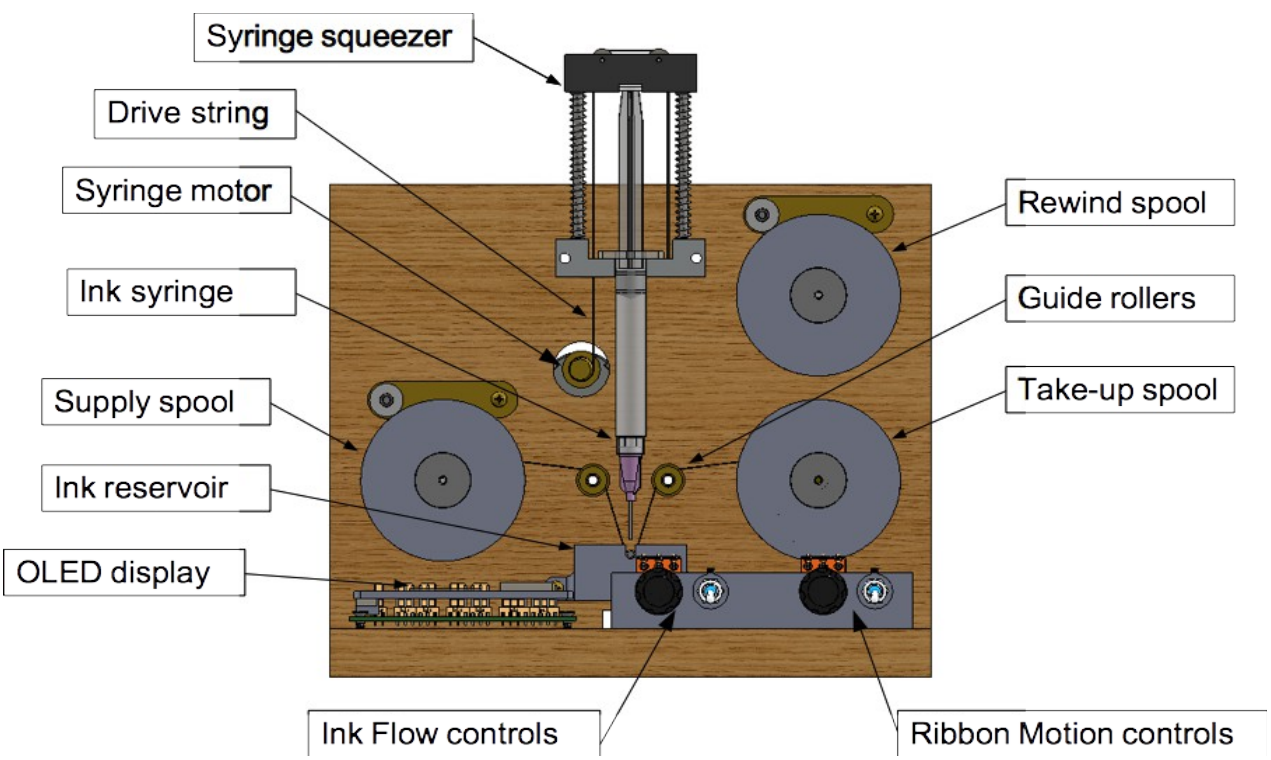
IBM model 029 Printing Card Punches generate punched cards for use by the IBM 1401 mainframe computer. Printing models of the 029 also print human-readable text along the top edge of the card. This text is created by a typewriter function that uses an inked ribbon. The ribbon width is much narrower (3/16" or 4.8 mm) than a standard typewriter ribbon, so replacement ribbons are generally unavailable. This creates a need to rejuvenate dry ribbons with fresh ink, hence the need for this ribbon ink application machine.

2 About Ribbons

The ribbons are not symmetrical end-to-end; the leading end features a metal strip that engages in the hub of the take-up spool, while the trailing end is permanently attached to the hub of the supply spool. A small grommet is placed near each end of the ribbon. These grommets interact with a direction reversal lever which causes the ribbon to reverse travel when the ribbon reaches the end of its travel. This means that inking a ribbon is a two step process: it must be first spooled off of the supply spool onto the take-up spool, then be rewound from the take-up spool back onto the supply spool. When re-inking is complete, the metal leading end will be on the outside of the spool.

3 Machine Layout

The machine components are shown in the figure below:



3.1 Supply Spool

This spool holds the dry ribbon to be re-inked. The spool is held in place by a knurled screw-on retainer screw. The spool is free to spin, but a Teflon bar on a swing arm provides a slight amount of drag on the spool rim to give the ribbon a slight amount of reverse tension.

3.2 Take-up Spool

This spool holds the freshly inked ribbon. It is driven by a geared stepper and pulls the ribbon through the guide rollers and the ink reservoir. The spool is held in place by a knurled screw-on retainer screw.

3.3 Rewind Spool

This spool holds the freshly inked ribbon as it is rewound back onto its original supply spool. The spool is held in place by a knurled screw-on retainer screw. The spool is free to spin, but a Teflon bar on a swing arm provides a slight amount of drag on the spool rim to give the ribbon a slight amount of reverse tension. After the ribbon has made a pass through the ink applicator, the full take-up spool is moved to the rewind spool position. In turn, the empty supply spool is moved to the take-up spool position. The ribbon is then wound onto the take-up spool.

3.4 Guide Rollers

Two guide rollers are located above the ink reservoir. They help control the position of the ribbon as it passes beneath the ink syringe. The ribbon is routed across the top of these rollers.

3.5 Ink Reservoir

The ink reservoir consists of a brass guide pin positioned to contact a sponge rubber block that is housed in a plastic tray. The ribbon passes below the pin, between the pin and the foam. Any excess ink that is not taken up by the ribbon is absorbed into the foam. The pin is removable from the rear of the unit, allowing the foam to be removed for cleaning.

3.6 Ink Syringe and Squeezer Mechanism

Ink to be deposited onto the ribbon from a precision ink dispenser. The dispenser consists of a syringe and a string drive syringe squeezer. A sliding bar contacts the end of the syringe plunger. The bar is pulled down by a string drive mechanism. A geared stepper motor turns a tensioning roller around which the drive string is wound. As the roller turns, the string pulls the sliding bar, forcing the plunger into the syringe. The speed of the motor directly controls the ink flow rate.

3.7 Ribbon Motion Controls

A three-way switch controls the motion of the ribbon motor. The center switch position is OFF, while the up and down positions correspond to forward and reverse directions respectively. Two directions are provided so that both clockwise and anti-clockwise winding is supported. A potentiometer adjusts the speed of the ribbon motor. The minimum speed is zero steps per second; the maximum speed is 500 steps per second.

3.8 Ink Flow Controls

A three-way switch controls the motion of the string drive motor. The center switch position is OFF, while the up and down positions correspond to upward and downward motion of the plunger respectively. A potentiometer adjusts the speed of the ink motor. The minimum speed is zero steps per second; the maximum operational speed is 30 steps per second. However if the potentiometer is turned to the fully clockwise position, the speed is set to 500 steps per second. This allows for rapid motion of the squeeze bar to facilitate the removal and reinstallation of the syringe for filling.

3.9 OLED Display

A small OLED display shows the state of the motor controls. A graphic icon at the bottom of the screen shows the STOP/FORWARD/REVERSE state of them motors. The current motor speed in steps per second is also shown for each motor.

4 Machine Usage

4.1 Filling the syringe

The syringe should be removed from the machine for filling. To do this, use the motor control to position the squeeze bar above the end of the syringe plunger. The syringe can then be pulled straight out of the machine. The syringe is then filled by drawing ink up into it from the supply bottle. A small condiment cup filled from the ink bottle has proven handy to contain the ink while it is being drawn into the syringe. Be sure to entrap as little air in the syringe as possible. Once the syringe is full, use the motor control to position the squeeze bar in such a way as the syringe can be loaded into the squeezer. It is necessary to manually pull the bar up as the drive string unwinds from its roller.

4.2 Loading the dry ribbon

Unscrew the knurled retainer screw from the end of the supply spool spindle. Place the dry ribbon on the spindle so that it it unrolls from the top of the spool, toward the guide rollers. Replace and tighten the retainer screw. Place an empty spool on the take-up spindle being careful to engage the drive boss into one of the holes on the spool flange. Replace and tighten the knurled retaining screw.

Pass the ribbon over the top of the guide rollers, then engage the metal ribbon end into the hub of the take-up spool so that the ribbon passes over the top of the take-up spool. Use the motor controls to advance take-up spool and seat the metal end. This will tension the ribbon a little bit. Pass the ribbon beneath the brass pin in the ink reservoir, between the pin and foam. The ribbon is now ready for inking.

4.3 Inking the ribbon

Start the ribbon moving forward slowly using the ribbon motor controls. When the ribbon is in motion, start the ink flow by adjusting the ink motor to provide a slow drip at the end of the syringe needle. Increase the ribbon speed to maximum. Adjust the ink flow to achieve the desired level of saturation. 2-5 steps per second was found to be a good range.

4.4 Rewinding the inked ribbon

When the full length of the ribbon has passed through the ink reservoir, stop the ribbon motion and then ink flow. Transfer the full spool to the rewind spindle. Transfer the empty spool to the take-up spindle. Start the ribbon motor in the appropriate direction and set the speed to maximum. When the ribbon has been fully wound on the take-up spool, the re-inking process is complete.

5 Maintenance

5.1 Cleaning and storage

The syringe should be emptied and washed when the inker is not in use. Use an appropriate cleaning agent that is compatible with the ink being used. The ink reservoir sponge should also be cleaned. Remove the brass pin from the rear of the unit to facilitate the removal of the foam.

5.2 Replacing the string

It is likely that the drive sting will require periodic replacement. The optimal string for this us surgical suture, although any strong, tightly-wound string will suffice. Remove the ink drive motor from the rear of the unit by removing its two mounting screws. This will allow access to the string drive roller. There is a small hole in the base of the threaded section of the roller. Pass the replacement string through this hole and secure it with a tight knot. Wind a few turns of string onto the roller before replacing the motor. Thread the string through the passages in the squeezer assembly. Be sure that the string passes cleanly over the top of the two small Teflon rollers in the squeeze bar. Tie a loop in the free end of the string, and put the loop over the head of the anchor screw. Use the ink motor controls to tension the string and syringe plunger.

6 Theory of Operation

The mechanical design of the machine is self-evident. The electrical design is also straightforward. Control is implemented by a Wemos ESP32 OLED Development Board. The software is written in the Arduino programming language.

The stepper motor drive sequence is implemented in software by the Arduino AcelStepper library. The operative step timing compute function runs in its own thread which is dedicated to one of the two ESP32 cores. This decouples the stepper motor timing from the main control and display loop. This is important because the OLED driver blocks its CPU during I2C communications. Without this threading and private CPU core, the stepper motor speeds would be unacceptably low. The winding drive is implemented by a pair of driver boards that use ULN2003 driver ICs.

The Wemos board does not bring out all of the available I/O pins from the ESP32 processor module. It was therefore necessary to wire pins directly from the module to their destinations on the connector perf board, bypassing the Wemos board. The device wiring is listed below.

Idx	Description	Signal Type	Wire Color	ESP32 GPIO	ESP32 Chip Pin	WROOM32 Module Pin	Wemos PCB Pin	OLED Display	Ribbon Motor Driver PCB Pin	Ink Motor Driver PCB Pin	Ribbon Speed Pot	Ink Flow Pot	Ribbon Control Switch	Ink Control Switch	J1	J2	J3	J4	J5	J6	J7
1	I2C SDA	Digital I/O	(internal)	5	34	29	24	SDA													
2	I2C SCL	Digital Out	(internal)	4	24	26	23	SCL													
3	Ribbon Motor Phase 1	Digital Out	YEL	12	18	14	19		1										3		
4	Ribbon Motor Phase 2	Digital Out	GRN	13	20	16	18		2									1			
5	Ribbon Motor Phase 3	Digital Out	BLU	14	17	13	20		3									2			
6	Ribbon Motor Phase 4	Digital Out	WHT	15	21	23	17		4									3			
7	Ink Motor Phase 1	Digital Out	YEL	16	25	27				1							3				
8	Ink Motor Phase 2	Digital Out	GRN	17	27	28				2						1					
9	Ink Motor Phase 3	Digital Out	BLU	18	35	30				3						2					
10	Ink Motor Phase 4	Digital Out	WHT	19	38	31				4						3					
11	Ribbon Speed Voltage	Analog	WHT/RED	36	5	4	1				Wiper										3
12	Ink Speed Voltage	Analog	GRN/RED	39	8	5	26					Wiper									4
13	Ribbon Motor Forward	Digital In	YEL	21	42	33							1							1	
14	Ribbon Motor Reverse	Digital In	GRN	22	39	36							3							2	
15	Ink Motor Forward	Digital In	BLU	23	36	37								1						3	
16	Ink Motor Reverse	Digital In	WHT	25	14	10	2							3						4	
17	Flash D1	Digital I/O	(internal)	8	33	22	4														
18	Flash CMD	Digital Out	(internal)	11	30	19	5														
19	Flash D0	Digital I/O	(internal)	7	32	21	6														
20	Flash SCK	Digital Out	(internal)	6	31	20	7														
21	Serial TX	Digital Out	(internal)	1	41	35	15														
22	Serial RX	Digital In	(internal)	3	40	34	16														
23	(unused)			26	15	11	3														
24	LED (LOW to flash)			2			21														
25	Reserved (LOW to boot)			0			22														
26	(unused)			16			25														
27	+3.3V 1						8				CW	CW									2
28	GND 1						9		GND	GND	CCW	CCW	2	2	1		1		1		1
29	+5V 1						10		+5V	+5V					2		2		2		
30	+3.3V 2						14														
31	GND 2						12														
32	GND 3						13														
33	+5V 2						11														

7 Software

A listing of the control software is shown below:

```
/*********
  Typewriter Ribbon Re-Inker
*********/

#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <AccelStepper.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define statHt 24

const int ribADCPin = 36;
const int inkADCPin = 39;

const int ribSwitchRev = 21;
const int ribSwitchFwd = 22;
const int inkSwitchRev = 23;
const int inkSwitchFwd = 25;

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

int ribSPS = 0;  //Ribbon motor steps per second
int inkSPS = 0;  //Ink dispenser steps per second

int ribState = 0; //Ribbon motor state: 0=stopped, 1=forward, -1=reverse
int inkState = 0; //Ink dispenser motor state: 0=stopped, 1=forward, -1=reverse

AccelStepper ribMotor(AccelStepper::FULL4WIRE, 12, 14, 13, 15);
AccelStepper inkMotor(AccelStepper::FULL4WIRE, 16, 18, 17, 19);

TaskHandle_t motorTask;

void setup() {
  Serial.begin(115200);

  // Start I2C Communication SDA = 5 and SCL = 4 on Wemos Lolin32 ESP32 with built-in SSD1306 OLED
  Wire.begin(5, 4);

  pinMode(ribSwitchFwd, INPUT_PULLUP);  // Define the motor control switch pins
  pinMode(ribSwitchRev, INPUT_PULLUP);
  pinMode(inkSwitchFwd, INPUT_PULLUP);
  pinMode(inkSwitchRev, INPUT_PULLUP);

  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C, false, false)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;);
  }

  //create a task that will be executed in the motorTaskCode() function, with priority 1 and executed on core 0
  xTaskCreatePinnedToCore(
    motorTaskCode,   // Task function.
    "motorTask",     // name of task.
    10000,           // Stack size of task
    NULL,            // parameter of the task
    1,               // priority of the task
    &motorTask,      // Task handle to keep track of created task
    1);              // pin task to core 1
  delay(500);

  display.setRotation(2);
  display.clearDisplay();
  display.setTextColor(WHITE); // Draw white text
  display.cp437(true);         // Use full 256 char 'Code Page 437' font

  display.drawRect(0, 0, 64, 64, WHITE); //Draw the screen border
  display.drawRect(64, 0, 64, 64, WHITE);
  display.drawRect(0, 64 - statHt, 64, statHt, WHITE);
  display.drawRect(64, 64 - statHt, 64, statHt, WHITE);

  display.setTextSize(1);      // Draw the static screen text
  display.setCursor(25, 3);
  display.print("INK");
  display.setCursor(78, 3);
  display.print("RIBBON");
  display.setCursor(21, 12);
  display.print("FLOW");
```

```
  display.setCursor(83, 12);
  display.print("SPEED");
  display.setTextSize(2);

  ribMotor.setMaxSpeed(500.0);
  ribMotor.setAcceleration(50.0);
  ribMotor.setSpeed(500.0);
  inkMotor.setMaxSpeed(500.0);
  inkMotor.setAcceleration(50.0);
  inkMotor.setSpeed(500.0);
}

void motorTaskCode( void * pvParameters ) {
  for (;;) {
    ribMotor.setSpeed(ribState * ribSPS);
    ribMotor.runSpeed();
    inkMotor.setSpeed(inkState * inkSPS);
    inkMotor.runSpeed();
  }
}

void loop() {
  if (digitalRead(ribSwitchFwd) == 0) {
    ribState = 1;
    ribFwd();
  } else if (digitalRead(ribSwitchRev) == 0) {
    ribState = -1;
    ribRev();
  } else {
    ribState = 0;
    ribStop();
  }

  if (digitalRead(inkSwitchFwd) == 0) {
    inkState = 1;
    inkUp();
  } else if (digitalRead(inkSwitchRev) == 0) {
    inkState = -1;
    inkDn();
  } else {
    inkState = 0;
    inkStop();
  }

  display.setTextColor(BLACK);
  display.setCursor(15, 23);
  display.print(inkSPS);
  display.setCursor(78, 23);
  display.print(ribSPS);
  ribSPS = map(analogRead(ribADCPin), 0, 4095, 0, 500);
  inkSPS = map(analogRead(inkADCPin), 0, 4095, 0, 30);
  if (inkSPS > 29) {
    inkSPS = 500;
  }
  display.setTextColor(WHITE);
  display.setCursor(15, 23);
  display.print(inkSPS);
  display.setCursor(78, 23);
  display.print(ribSPS);
  display.display();
}


void inkStop() {
  display.fillRect(1, 64 - statHt + 1, 64 - 2, statHt - 2, BLACK);
  display.fillRect(28, 70 - statHt, 5, 12, WHITE);
  display.fillRect(35, 70 - statHt, 5, 12, WHITE);
  display.display();
}

void ribStop() {
  display.fillRect(65, 64 - statHt + 1, 64 - 2, statHt - 2, BLACK);
  display.fillRect(64 + 28, 70 - statHt, 5, 12, WHITE);
  display.fillRect(64 + 35, 70 - statHt, 5, 12, WHITE);
  display.display();
}

void inkUp() {
  display.fillRect(1, 64 - statHt + 1, 64 - 2, statHt - 2, BLACK);
  //display.fillTriangle(28,70-statHt,28,70-statHt+12,28+12,70-statHt+6,WHITE);
  display.fillTriangle(26 + 8, 70 - statHt, 26 + 16, 70 - statHt + 12, 26, 70 - statHt + 12, WHITE);
  display.display();
}

void ribFwd() {
  display.fillRect(65, 64 - statHt + 1, 64 - 2, statHt - 2, BLACK);
  display.fillTriangle(64 + 28, 70 - statHt, 64 + 28, 70 - statHt + 12, 64 + 28 + 12, 70 - statHt + 6, WHITE);
  display.display();
```

```
}

void inkDn() {
  display.fillRect(1, 64 - statHt + 1, 64 - 2, statHt - 2, BLACK);
  //  display.fillTriangle(28,70-statHt+6,28+12,70-statHt,28+12,70-statHt+12,WHITE);
  display.fillTriangle(26 + 8, 70 - statHt + 12, 26 + 16, 70 - statHt, 26, 70 - statHt, WHITE);
  display.display();
}

void ribRev() {
  display.fillRect(65, 64 - statHt + 1, 64 - 2, statHt - 2, BLACK);
  display.fillTriangle(64 + 28, 70 - statHt + 6, 64 + 28 + 12, 70 - statHt, 64 + 28 + 12, 70 - statHt + 12, WHITE);
  display.display();
}
```